(1)

IMAGE UNDERSTANDING AND INFORMATION EXTRACTION

SEMIANNUAL SUMMARY REPORT OF RESEARCH FOR THE PERIOD April 1, 1977 - September 30, 1977

T. S. Huang and K. S. Fu

CO-PRINCIPAL INVESTIGATORS

School of Electrical Engineering

Purdue University

West Lafayette, Indiana 47907

TR-EE 77-41

November 1977

IMAGE UNDERSTANDING AND INFORMATION EXTRACTION

SEMIANNUAL SUMMARY REPORT OF RESEARCH

Contractor: Purdue University
Contract No.: MDA903-77-G-1
Effective Date of Contract: 1 April 1977
Contract Expiration Date: 31 March 1979
Amount of Contract: $650,000.00
Program Code Number: 7D30
Period of work covered: April - September 1977

Principal Investigator: T. S. Huang

Phone: 317-493-3361

K. S. Fu

Phone: 317-494-8825

This research was sponsored by the Defense Advanced Research Projects Agency under ARPA Order No.: 3412
Contract No.: MDA903-77-G-1
Monitored by Capt. David J. Brazil
AFAL/AAA
WPAFB, Ohio 45433

TABLE OF CONTENTS

IMAGE UNDERSTANDING AND INFORMATION EXTRACTION

ABSTRACT

This report summarizes the results of the research program on Image Understanding and Information Extraction at Purdue University. The report covers the period 1 April 1977 to 30 September 1977.

The objective of our research is to achieve a better understanding of image structure and to use this knowledge to develop techniques for image analysis and processing tasks, especially information extraction. Our emphasis is on syntactic decomposition and recognition of imagery based on scene analysis. It is our expectation that this research will form a basis for the development of technology relevant to military applications of machine extraction of information from aircraft and satellite imagery.

IMAGE UNDERSTANDING AND INFORMATION EXTRACTION

RESEARCH SUMMARY

The objective of our research is to achieve better understanding of image structure and to improve the capability of image processing systems to extract information from imagery and to convey that information in a useful form. The results of this research are expected to provide the basis for technology development relevant to military applications of machine extraction of information from aircraft and satellite imagery.

A block diagram of an Image Understanding System is shown in Fig. 1. We first consider the left side of the block diagram. After the sensor collects the image data, the preprocessor may either compress it for storage or transmission or it may attempt to put the data into a form more suitable for analysis. Image segmentation may simply involve locating objects in the image or, for complex scenes, determination of characteristically different regions may be required. Each of the objects or regions is categorized by the classifier which may use either classical decision-theoretic methods or some of the more recently developed syntactic methods. In linguistic terminology, the regions (objects) are primitives, and the classifier finds attributes for these primitives. Finally, the structural analyzer attempts to determine the spatial, spectral, and/or temporal relationships among the classified primitives. The output of the "Structure Analysis" block will be a description (qualitative as well as quantitative) of the original scene. Notice that the various blocks in the system are highly interactive. Usually, in analyzing a scene one has to go back and forth through the system several times.

Past research in image understanding and related areas at both Purdue and elsewhere has indicated that scene analysis can be successful only if we restrict a priori the class of scenes we are analyzing. This is reflected in the right side of the block diagram in Fig. 1. A world model is postulated for the

class of scenes at hand. This model is then used to guide each stage of the analyzing system. The results of each processing stage can be used in turn to refine the world model.

Research in image understanding at Purdue concerns with all aspects of the block diagram in Fig. 1. However, the emphasis will lie in the interaction between the processing stages (left side of Fig. 1), in the searching for suitable world models, and in the interplay between the world model and the processing stages.

Our research progress during the past six months is presented in this report.

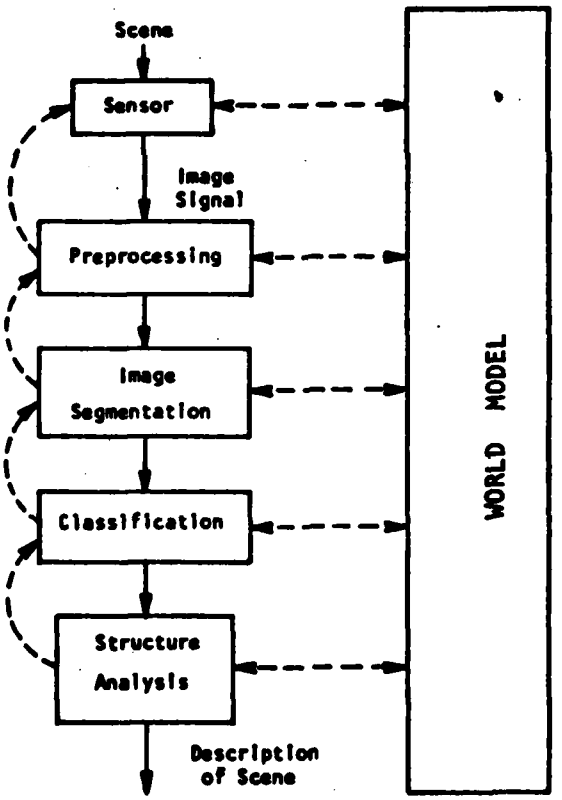


Fig. 1 An Image Understanding System

IMAGE SEGMENTATION - We are pursuing several approaches to image segmentation, including edge detection, region growing, and clustering. Some recent results on edge detection is reported in Salahi and Huang. The method consists of two steps. In the first step, a statistical test is used to detect edge points. This is followed by a second step where the edge points are selected and jointed to form smooth edges by a tree growing and pruning procedure.

IMAGE ATTRIBUTES - The method of describing shapes by Fourier boundary descriptors are now being applied to the recognition of three-dimensional aircrafts. Wallace and Wintz describes some very encouraging preliminary results.

IMAGE STRUCTURE - We feel that in many image analysis and recognition tasks, a purely syntactic approach is not adequate. An attempt to inject semantic considerations into the syntactic approach is presented by Tang and Huang. The work on the syntactic description of shapes continues; some results of airplane identification are described by Fu and You.

IMAGE RECOGNITION - Our work in this area concentrates on the use of contextual information to increase classification accuracy. Two reports are presented (Kit and Swain, and Yu and Fu).

PREPROCESSING - Many image processing problems call for two-dimensional recursive filters. In designing these filters, the test of stability is a key issue. O'Connor and Huang have developed several very efficient algorithms of testing stability.

APPLICATIONS - We are heavily involved in two mission-oriented projects. The first is the recognition of tactical targets in FLIR images which we are working on in collaboration with Honeywell (Carlton and Mitchell). The second is the tracking of moving targets in collaboration with the U.S. Army White Sand Missile Range (Mitchell).

IMPLEMENTATION - As we get into more mission-oriented projects where real-time processing is important, we begin to feel the inadequacy of general-purpose serially-processing computers. Therefore, we are studying computer architectures for image processing. Some considerations are presented by Fu and Keng.

EDGE DETECTION AND SMOOTHING

A. Salahi and T. S. Huang

INTRODUCTION

In [1] a test for edge detection was described. In this report we shall describe another algorithm which removes cracks and parasitic branches and fills missed edges.

DESCRIPTION OF SMOOTHING ALGORITHM

I A scanner scans the picture (according to some processing sequence) to find a starting point for an "edge follower" then stores the address of this point in R and pushes this address into some stack; say stack 2 which edge follower will use to pick up the starting point to follow the edge.

II Stack 2 will be popped up to give the root of a tree; then a tree will be induced into scattered edges around this point by using a breadth search method (B. S. M.) [2]. The expand part of B.S.M. will be described later. This tree will be expanded through depth $\leq$ limit; then the tree will be pruned and remaining leaves of tree will be pushed into stack 2 and part II will be repeated until stack 2 becomes empty.

III Scanner will be resumed from next point to R; if it finds another starting point, it stores that point into R and pushes its address into stack 2 and will go to (II); if it exhausts all the picture points then it will stop.

Pruning a tree

To prune a tree (Fig. 1) we use a very simple way:

1. Mark the leaves with a maximum depth of the tree with "-"
2. For each node with "-" change its "-" into "+" and put a "-" into its predecessor node (if it exists)
3. If there is no node with "-" in the tree, then remove all nodes without any "+" mark

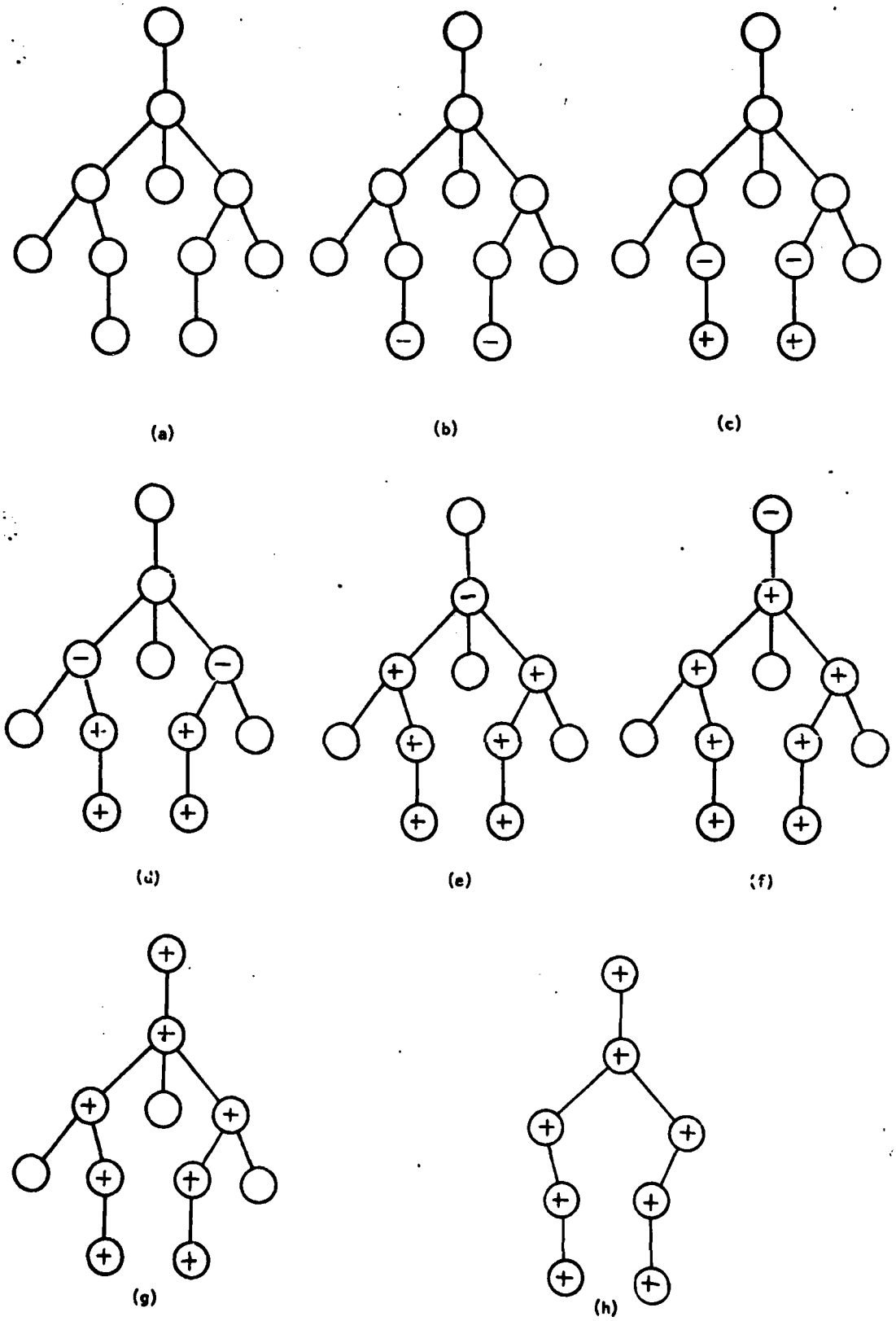


Fig. 1

PROCESSING SEQUENCE

Because edge smoothing through edge following is a sequential operation, the sequence of processing of different points (to find the starting point for an edge follower) has some effect although negligible on final results. By processing sequence we mean a one to one mapping from a two dimensional addressing into a linear addressing. Let I = {1.2....IRW}, J = {1,2,...ICN} K = {1,2,...IRW•ICN} and let $\underset{\sim}{\theta}$ = (IRW, ICN, i_0,j_0) where IRW, ICN are the number of rows and columns in digitized picture and (i_0,j_0) be the starting point. For simplicity let (i_0,j_0) = (1,1), then processing sequence will be

$$\underset{\sim}{\phi}_{\theta} : IXJ \rightarrow K$$

Some important examples are

$$\underset{\sim}{\phi}^1_{\theta} (i,j) = (i-1)\ ICN + j$$

$$\underset{\sim}{\phi}^2_{\theta} (i,j) = (j-1)\ IRW + i$$

$$\underset{\sim}{\phi}^3_{\theta} = \begin{cases} (i-1)\ ICN + j & i \text{ is odd} \\ i\ ICN - j+1 & i \text{ is even} \end{cases}$$

$$\underset{\sim}{\phi}^4_{\theta} = \begin{cases} (j-1)\ IRW + i & j \text{ is odd} \\ j\ IRW - i+1 & j \text{ is even} \end{cases}$$

Subroutine <u>next</u> in our algorithm uses $\underset{\sim}{\phi}^1_{\theta}$ to find the next point for processing. Subroutine <u>expand</u> in our algorithm generates the successor nodes of each node and will be discussed later.

DATA STRUCTURE

Because the technical description of our algorithm depends on data

Picture Points

A three bit data structure is sufficient to represent each point of picture

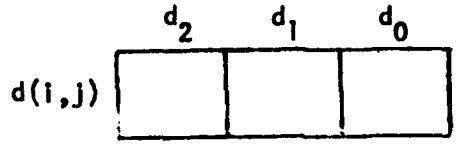


Fig. 2

$$d_0(i,j) = \begin{cases} 1 & \text{if there is a vertical edge between } (i,j) \text{ and } (i,j+1) \\ 0 & \text{otherwise} \end{cases}$$

$$d_1(i,j) = \begin{cases} 1 & \text{if there is a horizontal edge between } (i,j) \text{ and } (i+1,j) \\ 0 & \text{otherwise} \end{cases}$$

$$d_2(i,j) = \begin{cases} 1 & \text{point } (i,j) \text{ has already been reached by edge followers} \\ 0 & \text{otherwise} \end{cases}$$

One stack and one linear array namely stack 2 and A(ISS) will be used for edge follower and array A stores the tree in a layered form. Data structure for both stack 2 and array A are the same (Fig. 3.) After the tree was expanded and stored and pruned the content of array A will be similar to a new program; namely we can simulate a processor which accepts its instructions from array A and performs picture smoothing.

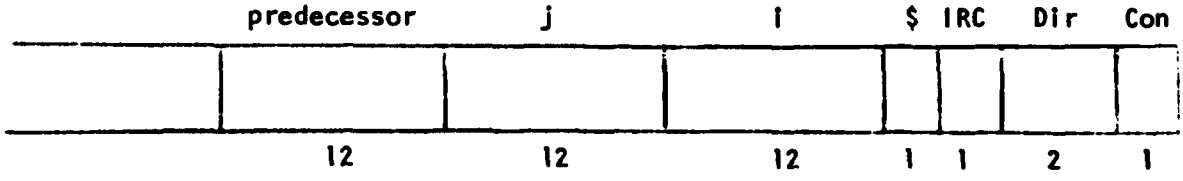


Fig. 3

(a) Con: Partial field Con is one bit; it shows edge approached point (i,j) through real (Con = 1) or imaginary edge (Con=0) Fig. 4.

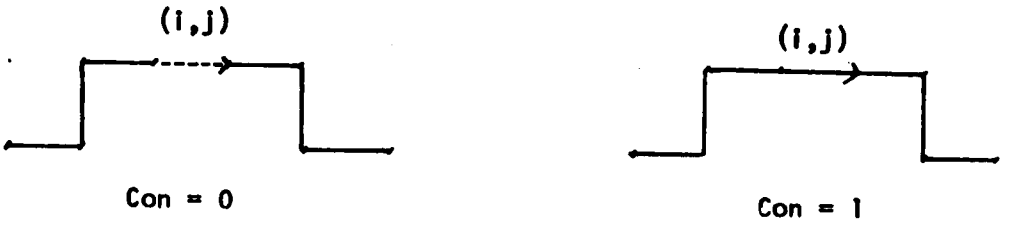


Fig. 4

(b) Dir: Edge follower can move in four directions (Fig. 5). Dir partial field is two bits and contains the direction edge follower used to move to point (i,j).

Fig. 5

(c) IRC is one bit

$$IRC = \begin{cases} 1 \text{ means this edge should remain in final image} \\ 0 \text{ means this edge should be removed} \end{cases}$$

(d) $ is one bit and it has been used as a separator of different layers of array A; all members of a layer have the same depth in a tree.

(e) (i,j) is the address of a point in a digitized picture.

(f) predecessor, points to predecessor of each node in Array A.

ALGORITHM FOR SMOOTHING

```
    Procedure Smooth (d, IRW, ICN, Limit)
    Integer Array d[1: IRW, 1: ICN], A[1:200], STACK 2 [1:200];
    Integer I, J, IR, JR, ICN, IRW, Limit, Z, PTR, PR2, Depth, ISW0;
    Switch V: = LL, LA1, LL, LA2, LL, LA3, LL, LA4, LA5, LL, LA6, LL, LA7,
                LL, LA8, LL;
    Label L1, L2, L3, L4, L5, L6, L7, L8;
Comment
    Algorithm has been written in Semi-Algol to be more descriptive;
Comment
    Array d is picture, A is linear array, STACK 2 is stack;
L1: (IR,JR): = (1,1);
L2: (I,J)  : = (IR,JR);
L3: (I,J)  : = Next (I,J);
    If (K,J) > (IRW, ICN)) Then Go to L8;
    If (d0(I,J) v d1(I,J)) ∧ d̄2(I,J) = 1 Then
        begin
          (IR,JR): = (I,J); d2(I,J): = 1; ISW0: = 0;
          If d0(I,J) = 1 Then [0,I,J,0,0,3,1] → STACK 2 else
                              [0,I,J-1, 0,0,3,0] → Stack 2
        end
          else Go To L3;
L4: depth: = 0; PTR: = 0; Stack 2 → A(1); $ → A(2); Z:= 2;
L5: PTR: = PTR + 1;
    If A(PTR) = $ Then
```

```
begin
  If PTR = Z Then
    begin
      Z: = Z-1;
      If ISWO ≠ 0 Then
           begin
             IRH: = 1; Go To L6
           end
             else
           begin
             IRH: = 0; Go To L6
           end
    end
           else
    begin
           depth: = depth + 1;
           If depth = Limit Then
              begin IRH: = 1; Go To L6 end else
              begin Z: = Z+1; A(Z) ← $; Go To L6 end
    end
end
  else
begin   expand (A, PTR, Z); Go To L5 end ;
```

```
L6:  for IRVS: = Z Step - 1 until 1 do
        begin
          If A(IRVS) = $ Then
            begin
              depth: = depth - 1;
              IRH: = 0
            end
              else
            begin
              If IRH = 1 Then
                begin
                  If depth = Limit Then
                    begin Stack 2 ← A(IRVS) end;
                  A(IRVS): = A(IRVS). OR. 8
                end;
              K1: = A(IRVS). AND. 8;
              If K1 ≠ 0 Then
                begin
                  IZX: = predecessor (A(PTR));
                  If IZX ≠ 0 Then
                    A(IZX): = A(IZX). OR. 8
                  end ;
              K2: = A(IRVS). AND . 15;
              Go To V(K2+1);
```

```
                    LA1: d(i,j): = d(i,j). AND . (-2); Go To LL;
                    LA2: d(i+1,j): = d(i+1,j) . AND . (-1); Go To LL;
                    LA3: d(i,j+1): = d(i,j+1) . AND . (-2); Go To LL;
                    LA4: d(i,j): = d(i,j) . AND . (-1); Go To LL;
                    LA5: d(i,j): = d(i,j) . OR . 2; Go To LL;
                    LA6: d(i+1,j): = d(i+1,j) . OR . 1; Go To LL;
                    LA7: d(i,j+1): = d(i,j+1) . OR . 2; Go To LL;
                    LA8: d(i,j): = d(i,j) . OR . 1; Go To LL;
                    LL:
                end
            end;
L7:  If Stack 2 is empty Then Go To L2 else
        begin
                    ISW0: = 1; Go To L4
        end;
L8:
end
```

expand (See Fig. 6)

Objective: This algorithm generates the successor nodes of each node in Breath Search Method.

Let S = (i,j) be a point in which edge entered with direction x; from S it can go at most to 3 neighboring points A, B, C.

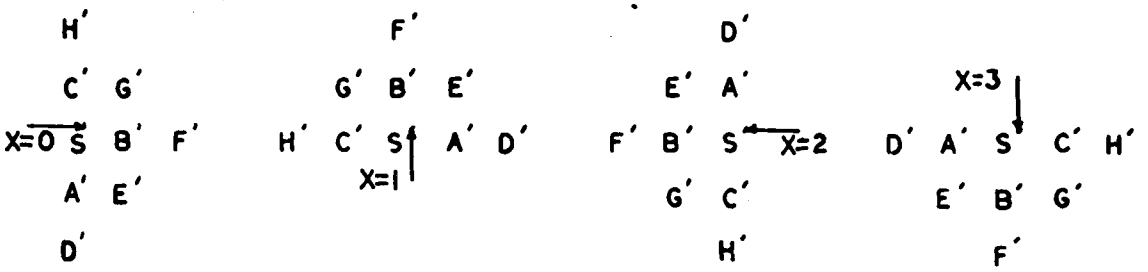


Fig. 6

```
begin
    d2(S): = 1;
    If SA exists Then begin
        If d2 (A) = 1 Then remove SA else [0,A,0,0, < x+3 >*, 1] → Stack 2
                    end
                        else
                    begin
        If d2 (A) = 0 Then begin
                    If AD or AE exist Then [0,A,0,0,< x+3 >, 0] → Stack 2
                        end
                    end;
    If SB exists Then begin
        If d2 (B) = 1 Then remove SB else [0,B,0,0, < x >, 1] → Stack 2
                    end
                        else
                    begin
        If d2 (B) = 0 Then begin
                    If BE or BF or BG exist Then [0,B,0,0,x,0] → Stack 2
                        end
                    end;
    If SC exists Then begin
        If d2 (C) = 1 Then remove SC else [0,C,0,0, < x+1 >, 1] → Stack 2
                    end
                        else
                    begin
```

```
                    else
                    begin
        If d2 (B) = 0 Then begin
                    If BE or BF or BG exist Then [0,B,0,0,x,0] → Stack 2
                        end
                    end;
    If SC exists Then begin
        If d2 (C) = 1 Then remove SC else [0,C,0,0, < x+1 >] → Stack 2
                    end
                        else
                    begin
            If d2 (C) = 0 Then begin
                    If CH or CG exists Then [0,C,0,0, < x+1 >] → Stack 2
                        end
                    end
end
```

$< x >$ means $< x > \equiv x \bmod 4$ and $0 \leq < x > < 4$

As we already mentioned the final contents of Array A can be considered as a new program for smoothing.

Let our OP Code consist of two bits IRC & Con Then

IRC	Con	Meaning	
0	0	Nop	no change
0	1	REMOVE	Remove the edge
1	0	Fill	Fill the edge
1	1	Nop	no change

The general format of instruction can be in the following way.

OP	Dir	Address

for example 01 11 6,5 means remove edge which entered to point (6,5) with direction 3

To show the effect of algorithm let Fig. 7 be the result of edge detection. Table 1 shows Array A before pruning. Table 2 shows Array A after pruning. Fig. 8 shows Fig. 7 after smoothing. Contents of A can be translated into the following program.

PC	OP	dir	Address
22.	Nop		
21.	Nop		
20.	Fill	0	7,6
19.	Nop		
18.	Nop		
17.	REMOVE	2	7,3
16.	Nop		
15.	Nop		
14.	Remove	3	6,3
13.	Nop		
12.	Fill	3	6,4
11.	Nop		
10.	Nop		
9.	Remove	0	4,5
8.	Nop		
7.	Nop		
6.	fill	3	4,4
5.	Nop		
4.	Nop		
3.	Remove	2	3,2
2.	Nop		
1.	Nop		

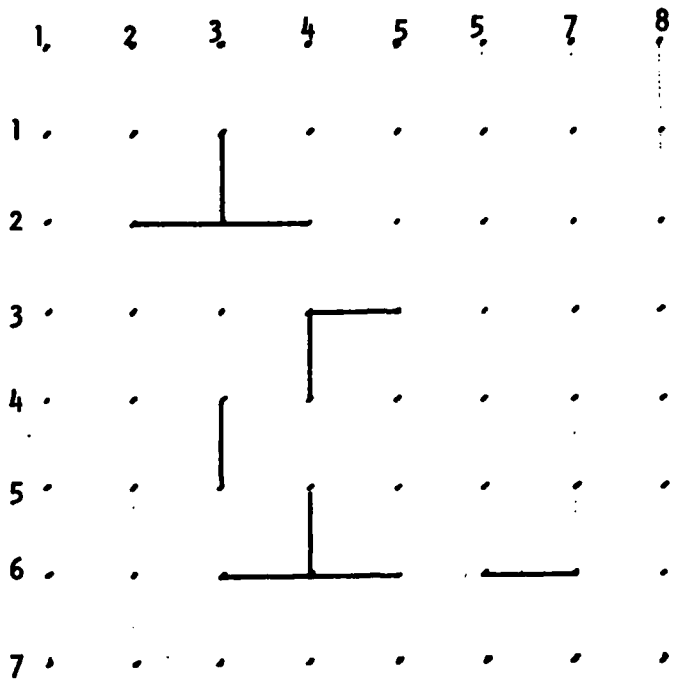


Fig. 7

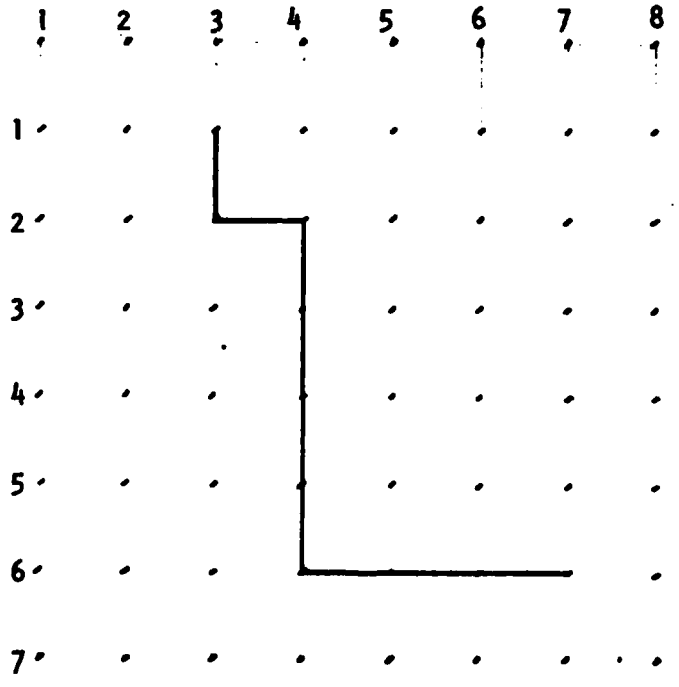


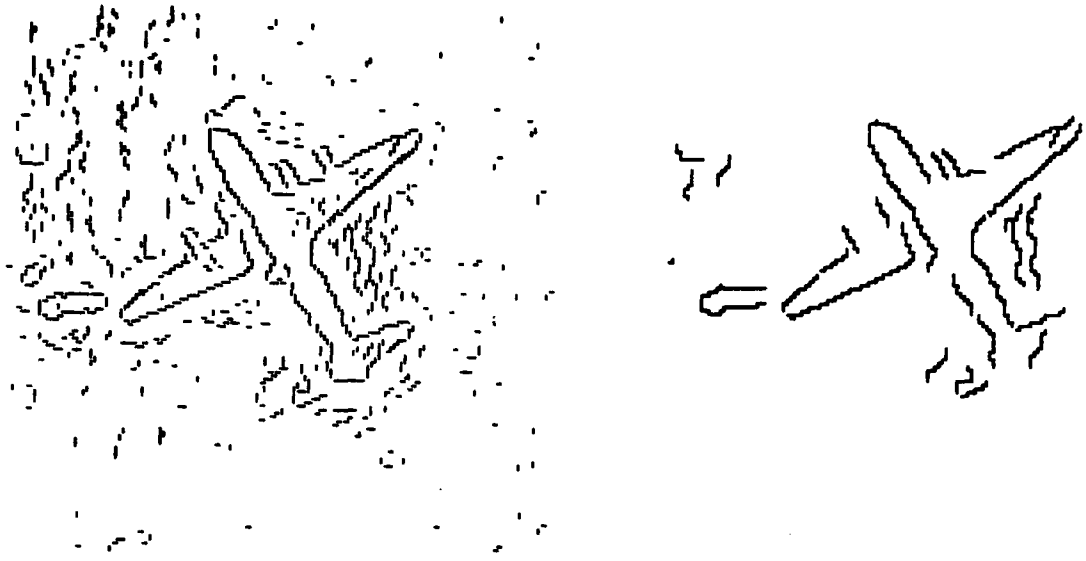
Fig. 8

Table 1

1		0	3	3	0	0	3	1
2					1			
3		1	3	2	0	0	2	1
4		1	3	4	0	0	0	1
5					1			
6		4	4	4	0	0	3	0
7					1			
8		6	5	4	0	0	3	1
9		6	4	5	0	0	0	1
10					1			
11		8	5	3	0	0	2	0
12		8	6	4	0	0	3	0
13					1			
14		11	6	3	0	0	3	1
15		12	7	4	0	0	3	1
16					1			
17		15	7	3	0	0	2	1
18		15	7	5	0	0	0	1
19					1			
20		18	7	6	0	0	0	0
21					1			
22		20	7	7	0	0	0	1

Table 2

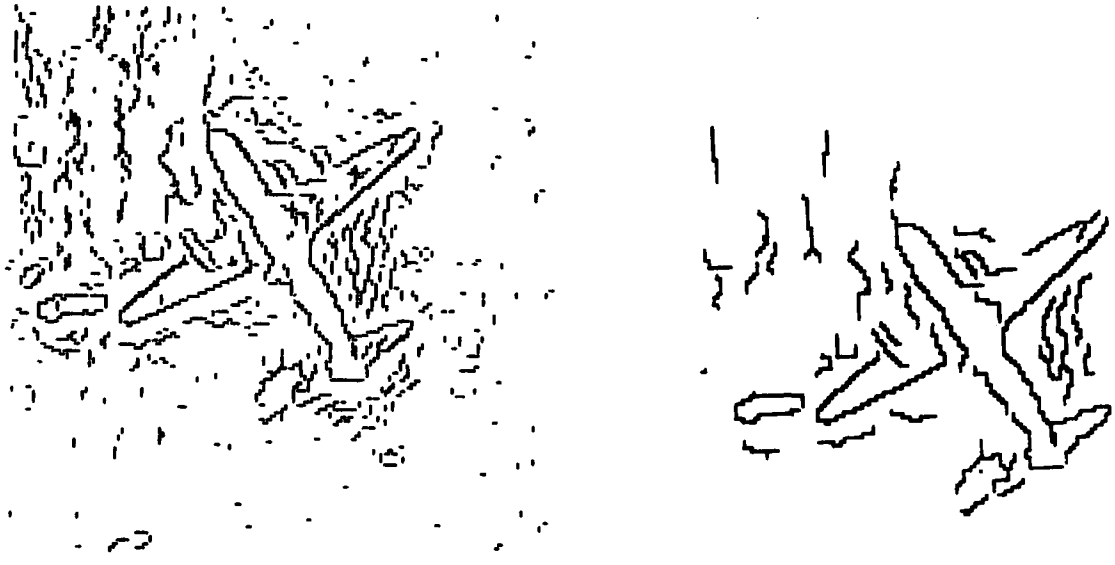
1		0	3	3	0	1	3	1
2					1			
3		1	3	2	0	0	2	1
4		1	3	4	0	1	0	1
5					1			
6		4	4	4	0	1	3	0
7					1			
8		6	5	4	0	1	3	1
9		6	4	5	0	0	0	1
10					1			
11		8	5	3	0	0	2	0
12		8	6	4	0	1	3	0
13					1			
14		11	6	3	0	0	3	1
15		12	7	4	0	1	3	1
16						1		
17		15	7	3	0	0	2	1
18		15	7	5	0	1	0	1
19					1			
20		18	7	6	0	1	0	0
21					1			
22		20	7	7	0	1	0	1



(a) (b)

Idis = 1 d = 10
Limit = 7 TFJ 20

Fig. 9



(a) (b)

d = 10 Limit = 6
TFJ 399

Fig. 10

RESULTS (Figs. 9 and 10)

Let us assume Algorithm 1 to be the algorithm which we used for statistical edge detection [1] and Algorithm 2 the present algorithm for smoothing.

Let $P\{|t| \leq t_r\} = 1-\alpha$ and d be the maximum depth of tree which edge follower uses. Then

Part (a) of each figure is the result of Algorithm 1 under Limit = t_r

(b) shows (a) after using algorithm 2

REFERENCE

[1] A. Salahi and K. S. Fu, "A Decision-Theoretic Approach to Edge Detection," Purdue Univ. Tech. Rep., TR-EE 77-16, March 1977.

[2] Nils J. Nilsson, Problem Solving Method in Artificial Intelligence, McGraw-Hill, N.Y. 1971.

SHAPE INFORMATION EXTRACTION

T. Wallace and P. A. Wintz

INTRODUCTION

In our last final report [1], a practical shape information extraction algorithm was presented which detected the presence of an airplane in an actual photograph. The classification was made using Fourier descriptors, which were computed from the boundary of the plane. The boundary was in turn located by the BLOB region detection algorithm. In our last quarterly report [2], we presented several approaches to improving and extending this algorithm. This report describes certain theoretical results of interest in connection with three dimensional shape recognition using Fourier Descriptors. In addition, preliminary experimental results are presented.

FOURIER DESCRIPTORS

The Three-dimensional Problem

The Fourier descriptor (FD) algorithm is a very powerful technique for classifying shapes, but work to date has concentrated on the two dimensional recognition problem. Certain properties of FDs facilitate their use in three dimensional object detection, providing both an identification of the unknown object and an estimate of its orientation relative to the camera. Specifically, it has been shown [3] that averaging the FDs of two different shapes (frequency domain) yields a FD which will inverse transform to a shape which appears to be an "average" contour, intermediate in shape between the two original contours. The data base which must be stored to represent a three-dimensional object may be reduced by using fewer projections, and "interpolating" between them in the frequency domain. This approach also enables a more accurate estimation of the actual orientation in space of the object relative to the camera. Previous work on this estimation problem [4] has

assumed that the orientation of the unknown object is that of the nearest reference projection. This evidently limits estimation accuracy to the resolution of the reference projection set.

Interpolation and Sampling Error

One theoretical problem concerns the sample spacing used in sampling a time domain contour. As explained in our last final report, a uniform sampling strategy is employed in the present algorithm, in order to facilitate analysis of a wide variety of shapes. While non-uniform sampling can result in faster convergence of a FD [3], there are obvious complications involved in defining such a sampling strategy for general shapes. When operations on FDs are made in the frequency domain, there is no guarantee that the resulting time domain representation will have uniform sampling. Hence, even if an unknown contour appeared identical to an "average" contour computed by using linear combinations of known FDs, different sample spacing could result in some difference between the two FDs.

Consider the case in which an unknown projection lies directly between two library projections. We postulate that some linear combination of the library projections A(1) and A(2) will give a good approximation to the unknown projection X:

$$X = k(1) \cdot A(1) + k(2) \cdot A(2)$$

where the k(i) are scalers with $k(1) + k(2) = 1.0$

and the A(i) and X are FDs. Due to linearity, we can say that this frequency domain interpolation FD X has a transform x which can also be computed by interpolating between space domain vectors using the same coefficients.

$$x = k(1) \cdot a(1) + k(2) \cdot a(2)$$

where the a(i) are the IDFTs of the A(i). It is now obvious that the point spacing of x need not be uniform, and it is also easy to compute the point

density error for some test interpolations. We merely compute the vector x in the space domain using (2), then resample that vector uniformly to get some x′, and finally compute the m.s. distance between x and x′. Experiments with two shapes whose normalized FDs (NFDs) had a distance of about .3 (a distance less than .1 is generally used as a classification threshold) showed a point density error 190 to 200 times less than the distance between the original NFDs, with k(1) = k(2) = .5. The number of points used in the space domain vectors had a slight effect on the error, with more densely sampled vectors producing slightly less error. It can be concluded that this problem should not have a noticable effect on the algorithm, since in practice, adjacent projections can be expected to have a NFD distance less than .3. This fact should further minimize the point density error.

The Estimation Problem

In the example discussed above, a projection was assumed to lie directly between two library projections, and the experiment was performed accordingly. In general, however, any random projection will not lie on the grid defined by a library of projections. Hence more than two library projections must be used to perform the estimation. If a rectangular grid of projections is used, it would seem reasonable to do the estimation based on four library projections, but consider instead the general case of estimating an n-vector X(k) as a linear combination of m n-vectors ($Y_i(k)$, $1 < i < m$:

$$\hat{X}(k) = \sum_{i=1}^{m} a_i \, Y_i(k)$$

subject to the restriction that

$$\sum_{i=1}^{m} a_i = 1.0$$

The following derivation yields an expression for the optimum linear m.s. estimate of X.

$$\hat{X}(k) = \sum_{i=1}^{m-1} a_i \, Y_i(k) + (1 - \sum_{i=1}^{m-1} a_i) \, Y_m(k)$$

$$\hat{X}(k) = \sum_{i=1}^{m-1} a_i [Y_i(k) - Y_m(k)] + Y_m(k) = \sum_{i=1}^{m-1} a_i \, D_i(k) + Y_m(k)$$

where

$$D_i(k) = Y_i(k) - Y_m(k)$$

The total m.s. error is given by:

$$E = \sum_{k=1}^{n} (X(k) - \hat{X}(k))^2$$

$$= \sum_{k=1}^{n} \{ X^2(k) - 2 \, X(k) \, [\sum_{i=1}^{m-1} a_i \, D_i(k) + Y_m(k)] +$$

$$[\sum_{i=1}^{m-1} a_i \, D_i(k) + Y_m(k)]^2 \}$$

Differentiating with respect to the a_i, setting each resulting equation equal to zero, and rearranging:

$$\sum_{k=1}^{n} 2[X(k) - Y_m(k)] \, D_i(k) = \sum_{j=1}^{m-1} \sum_{k=1}^{n} a_j \, D_i(k) \, D_j(k) + a_i D_i^2(k)$$

$$\sum_{k=1}^{n} 2[X(k) - Y_m(k)] \, D_i(k) = \sum_{j=1}^{m-1} a_j [\sum_{k=1}^{n} D_i(k) \, D_j(k) + a_i D_i^2(k)]$$

For $1 \leq i \leq m-1$.

This expression is a set of m-1 equations in m-1 unknowns which can be solved for the unknowns a_i. a_m is then computed from

$$a_m = 1.0 - \sum_{i=1}^{m-1} a_i$$

Data Reduction

The time required to execute the above estimation algorithm is dependent on the dimension (n) of the vectors. It is thus desirable to reduce the dimension of the vectors as far as possible without degrading the classification performance. Equally important is the problem of storage of library data representing a three-dimensional object. Previous FD classification results have indicated that there is no advantage in using more than 30 (complex) coefficients. In fact, quite good results have been obtained with only 14, although there was a slight degradation in performance when compared with 30.

The obvious approach is to estimate the autocorrelation matrix or covariance matrix of the data, and find the eigenvalues and eigenvectors which provide optimal data compression. There can be some difficulty in computing eigenvalues and eigenvectors of a 60 by 60 matrix. (Our feature vector consists of 30 complex coefficients.) One way to reduce the size of this matrix is to convert the data to 30 real coefficients. There are two ways that this might be done. The most obvious way is to simply take the magnitudes of the FD coefficients, since classifications based on magnitude information alone have been shown to be quite effective. However, if the data has bilateral symmetry, the associated NFDs should automatically be real. Even if the data does not have this symmetry, the normalization procedure tends to minimize the magnitudes of the imaginary parts of the NFD, and correspondingly minimize their contribution to the classification. Hence virtually all the information can be preserved by simply taking the real part of each NFD coefficient. This is the approach that was used in the experiment described below. The eigenvalue/eigenvector computation was performed using just the real parts of the data, but the transformation matrix thus obtained was

applied to both the real and imaginary parts of the data, yielding a complex vector in the transformed space.

Using the covariance matrix approach results in maximum compression of the data, but the time to compute the transformed vectors is slightly increased due to the necessity to subtract the mean vector. In cases where extreme data reduction is required, this might be the best approach, but since the present algorithm requires keeping virtually all the information, the advantages should be slight. In other words, the information contained in five to ten coefficients of the transformed space should be essentially the same regardless of which method is employed. On the other hand the autocorrelation matrix method is slightly simpler and lends itself to a more intuitive interpretation.

The basic FD classification procedure can be viewed as reducing a shape to circular or elliptical basis shapes of various frequencies. The transformed FD classification procedure makes use of a more efficient set of basis shapes. Recall that the normalization procedure for FDs results in a dc component defined to be zero ($a(0) = 0$), and a fundamental frequency component defined to be unity ($a(1) = 1.0$). Since these coefficients would make the autocorrelation or covariance matrix singular, they are not included in the original data vectors. They of course contribute nothing to the classification process. In order to see what sort of basis shapes we are classifying with, it is only necessary to inverse transform each transformed coordinate in turn, and then insert a fundamental frequency of some appropriate magnitude. We obtain the FD of some shape, and performing an inverse FFT shows us what it looks like.

The experiment originally performed to test the FD algorithm was repeated to test the data reduction procedure. Recall that 20 low resolution airplane

silhouettes after the FDs were calculated and normalized. Using a mean square distance criterion and 30 coefficients, the classification accuracy was 95%. After computing eigenvectors and eigenvalues of the autocorrelation matrix of the NFDs, and making the appropriate linear transformation, a classification using four transformed coefficients achieved the same figure of 95% classification accuracy. Figures 1-4 show the basis shapes associated with these four coefficients.

THE 3-D ALGORITHM

An experiment was performed in which unknown aircraft outlines were identified and their orientation in space estimated using the above results. First, a set of aircraft was synthesized using a graphics approach. Three-dimensional approximations were constructed for six different aircraft, a Mirage, a Mig, a Phantom, an F104, an F105, and a B57. Figure 5 shows representative images generated by this program. These three dimensional images were then rotated through appropriate angles to create a library of projections. The program was first given the library, and then given randomly selected orientations to identify.

The experiment of Dudani et al [4] was very similar, but several important differences should be noted. First, the data used by Dudani was constructed using model aircraft and a television camera hookup. It might appear that this is a more realistic approach, as well as a more demanding experiment than one using graphically generated data. However there are two problems with this method which the graphical method avoids. First, the resolution of the mechanical mount used by Dudani was 5 degrees. This represents an error in data generation which is avoided by the more exact graphics approach. Second, since the camera is a finite distance from the model, parallax problems affect the images, making the camera image different

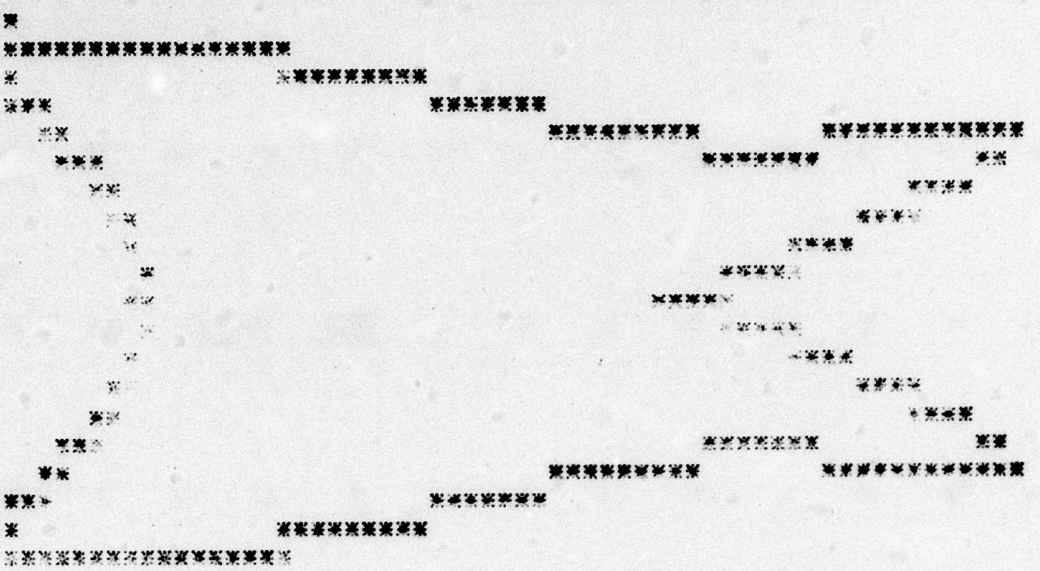


Figure 1

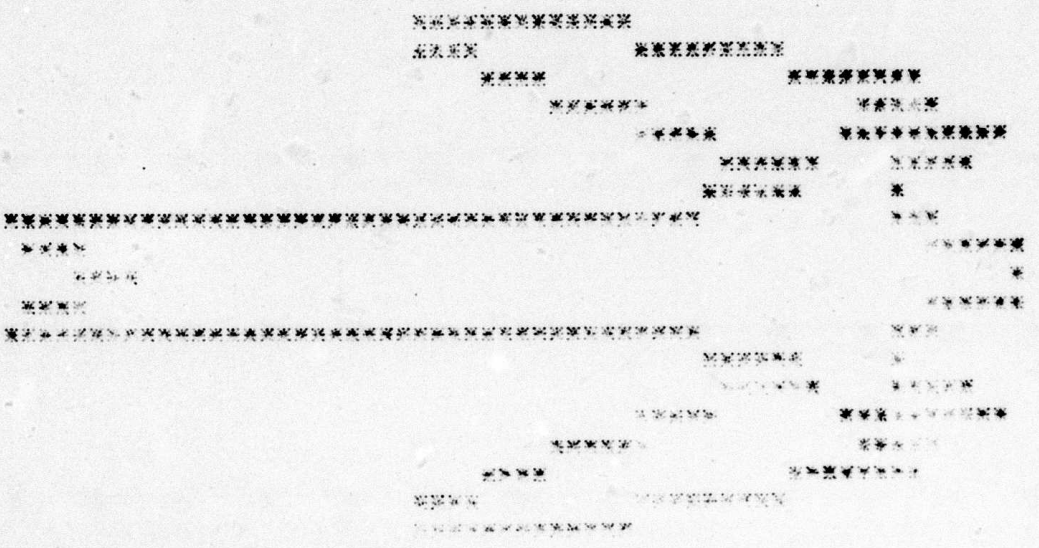


Figure 2

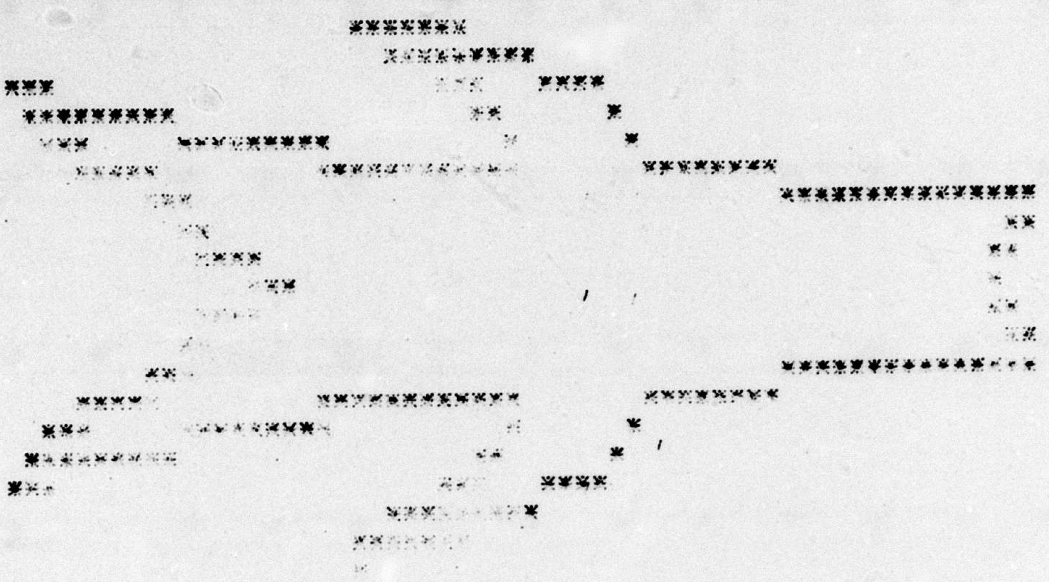


Figure 3

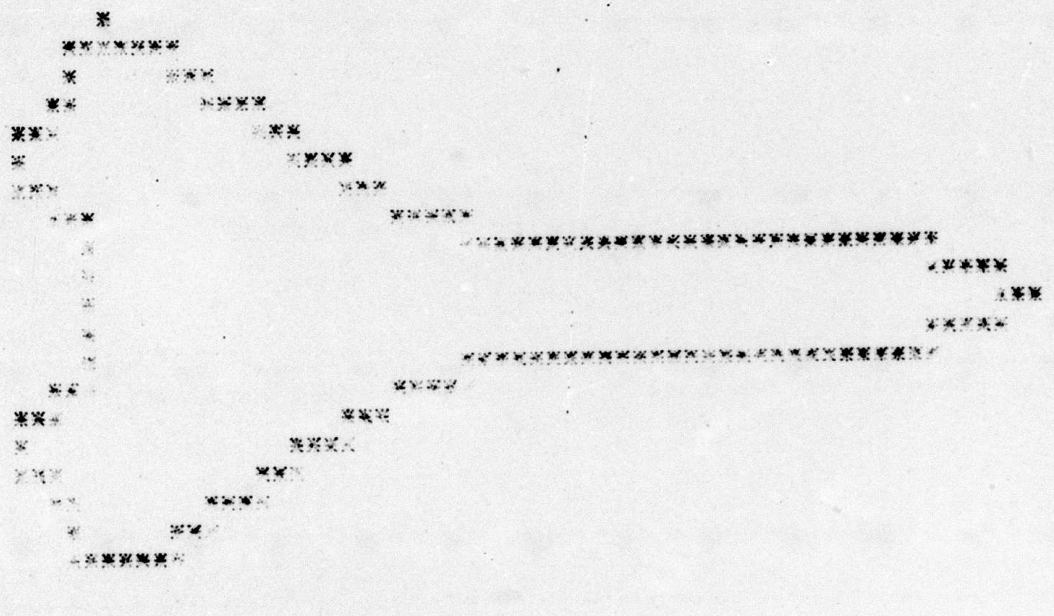


Figure 4

from the image received from a long distance. Since most practical photographs of actual aircraft in flight would be taken at a large distance, this error is undesirable.

In addition to the above considerations, it would probably be easier to use graphics techniques in a practical system, since accurate graphical representations constructed from blueprints would probably generate library data faster and more accurately than model-tv camera setups.

A major advantage of Dudani's approach is the accuracy of the aircraft shape. Our graphics program approximates each plane by using about 50-100 (geometric) planes. A more elaborate program could generate an arbitrarily accurate representation of each aircraft, with corresponding increase in computation time. The present data gives a reasonably good approximation to each aircraft, although the small detail is lacking. The effect on the classifications is probably to increase their difficulty, since certain minor features are missing. On the other hand, the data can probably be represented by fewer projections, due to the reduced complexity.

The basic problem considered by Dudani was classifying unknown aircraft images oriented at 5 degree intervals in a 140 degree by 90 degree sector. Each aircraft was represented by a library of moment feature vectors computed from 551 projections within this sector. The classification was performed by computing distances from a moment feature vector of the unknown image to the moment feature vectors of the library images, and then classifying using a distance-weighted k-nearest neighbor rule. Note that the images used by Dudani did not contain any mirror image pairs, and hence are obviously not all the images which can be theoretically recognized. In fact, a little reflection will convince one that if an object has enough assymetry, it can be recognized at any angle at all, and that angle identified. In the case of

aircraft, there is generally bilateral symmetry, but this does not necessarily greatly limit the set of angles which can be theoretically recognized.

Our algorithm recognizes aircraft outlines taken from a sector of 180 by 180 degrees, i.e. a hemisphere. Note also that if the angles near the front view and rear view of the aircraft are deleted, the problem is much easier, since the shapes vary much more radically when large surfaces are viewed almost edgewise. Dudani's consideration of only 140 degrees reduces this problem. Our algorithm also recognizes random projections. There is no quantization of random projections corresponding to Dudani's 5 degree increments. Finally, the first version of our algorithm uses only 99 projections to represent an aircraft over the hemisphere, which represents a density of projections 14.3 times less than that used by Dudani.

The actual classification program proceeds as follows. First, the library of projections is computed, and the NFD of each projection is computed. The autocorrelation matrix of the NFDs is computed, and an eigenvalue-eigenvector transformation reduces the data dimensionality from 30 complex numbers to 5 complex numbers. The real parts of the complex numbers are used to compute the autocorrelation matrix, but the complex parts of the transformed coefficients are kept to assist in the classification.

Next the m.s. distance from a given unknown contour to each library vector is computed. The distance to the nearest library contour is saved as the current best estimate of the minimum m.s. distance achievable. The projections adjacent to the nearest library projection are investigated by the m.s. estimation algorithm described above in an attempt to interpolate between the library projections.

The interpretation of the estimation coefficients returned by the estimation subroutine is somewhat heuristic, and goes something like this.

This routine is constrained to return four numbers whose sum is 1.0. It often happens that two of the vectors being used to estimate the unknown contour are not very similar to the unknown contour, but are quite similar to each other. In this case, the estimation coefficients are of similar magnitudes and opposite sign, such as 2.0 and -2.05. What the estimation algorithm is doing is using the difference vector to help generate the optimum m.s. estimage of the unknown vector. We of course do not want to allow this kind of estimation, since it is inconsistent with our theory of interpolation of FDs. Another thing which is commonly observed when the unknown vector differs from the library vectors being used to estimate it is a set of large positive and negative estimation coefficients being returned. This again just tells us that we cannot expect to find a reasonable interpolated FD in the sector determined by that set of library projections.

The heuristic solution to these effects is as follows. First, we quit looking in a particular sector if the estimation coefficients returned are too large in magnitude. The algorithm is not very sensitive to this magnitude and 1.5 to 2.0 is usually used. Also, if two coefficients sum to a small number (.1), but have relatively large magnitudes (>.5) they are assumed to be cancelling coefficients, and are deleted from the estimation set. The estimation process is then repeated with the remaining two vectors being used to estimate the unknown set. Similarly, any negative coefficients are deleted from an estimation, and the remaining two or three are used to repeat the estimation process. When an estimation of the unknown vector in terms of two, three, or four adjacent library projection vectors is achieved in which all the coefficients lie between zero and one, the distance is compared to the minimum distance achieved so far. If the new distance is less, the minimum distance is updated.

This process may be repeated for the k nearest library projections, where k is optional, and is generally in the range of 4-10. If the distances to the nearest k projections are approximately equal, the full k projections will be investigated. However, projections whose distances are more than 1.5 to 2.0 times greater than the minimum distance are not investigated. Each library projection has one, two, or four sectors surrounding it which must be investigated by the estimation subroutine. (If the sector is in the middle of the library set, there are four, if it is on the border, there are two, and if it is in a corner, there is one.) After the desired number of possible sectors are investigated, there are two possible procedures. The estimated orientation is taken to be that of the original nearest library vector, if the estimation fails to improve on this distance. If the estimation procedure is successful, the orientation is computed by multiplying the orientations of the vectors used in the estimation by their appropriate coefficients.

Results to date using 6 aircraft, and classifying 50 unknown images for each one show a classification accuracy of 80% overall. The classification accuracy is about 72% if the estimation routine is not used.

It is expected that this figure can be improved by making use of the real and imaginary parts of the data when computing the eigenvector transformation matrix. Also, the spacing of library projections used in this experiment is non-uniform in an attempt to increase the Fourier space distance uniformity of the projection set, but it is not optimum. Finally, it may be necessary to increase the density of projections somewhat to bring the classification accuracy up to that achieved by Dudani. The current attempt to get by with a projection density 14.3 times lower than Dudani may be too optimistic.

Given a chain code representation of the outline of an aircraft projection, the normalized FD is computed in about 2.76 sec. This time includes

an FFT which is of length 512 or 1024, depending on the length of the particular contour. This normalized FD is then classified and its orientation estimated in about 2.38 sec. These times are for a 11/45 with floating point hardware. The program itself is written in Fortran and is a research tool rather than a highly efficient implementation of the algorithm.

SHAPE DETECTION ALGORITHMS

The Blob Algorithm

The BLOB program for segmenting pictures has been used successfully in classification of multispectral data and more recently in locating aircraft in ordinary photographic data. The regions located by the BLOB program are similar in both mean and variance. In order to improve BLOB performance using single channel data, we have been reexamining the pixel group statistics used by BLOB with regard to their applicability to our data.

Mean information is obviously very important, and correlates well with distinct regions as defined by human observers. The use of variance information is not as obvious a procedure, and we have recently been examining photographic data with a program which replaces each pixel group by its standard deviation, simulating with BLOB "sees" when it does its variance test. There are two classes of pictorial data which are evident when variance information is examined--those which were created by imaging processes with sufficient dynamic range to represent the highest gray levels, and those which "clip" at high gray levels.

In the latter case, it is obvious that variance information is very much a function of the fraction of the image which was too bright for the imaging system. In the former case, the surprising result has been obtained that, with our aerial photographs, the variance information is largely a duplication of the mean information. One might expect that some regions would have

high brightness and low variance, and some regions low brightness and high variance, but this is not what generally occurs. In fact, low brightness regions almost invariably have low variance, and high brightness regions which may appear completely uniform, as a painted airplane body, have relatively high variance.

This situation may be partially explained by the logarithmic characteristic of the human eye. At low gray levels, variance information is easily seen as the eye can discriminate between closely spaced shades quite well here. At high gray levels, variance information is obscured by the insensitivity of the eye to gray level variations of even tens of gray levels. (Our pictures are displayed with 256 gray levels.)

After taking the logarithm of a picture, the quality often does not appear to be impaired to the human observer, but the standard deviations of pixel groups appear to be almost random, with a few slightly higher variance regions which still correspond to the higher mean regions. Taking the logarithm again often does not result in serious degradation of the picture, but the variance information appears to be random noise!

We have not discovered a function of the gray levels of two by two pixel groups which contains useful information for use in segmenting the picture, and which is not largely a repetition of the mean information. One possibility is to retain the geometry (contour-tracing algorithm) of the BLOB program, and remove the variance test from the program. Using this version of BLOB perhaps interactively with the growing and shrinking algorithms described in our last quarterly report may provide the best overall performance yet.

REFERENCES

[1] T. Wallace and P. A. Wintz, "Fourier Descriptors for Extraction of Shape Information," Final Report of Research for the period Nov. 1, 1975 -Oct. 31, 1976, ARPA Contract No. F 30602-75-C-0150.

[2] T. Wallace and P. A. Wintz, "Shape Information Extraction," TR-EE 77-16, School of Electrical Engineering, Purdue University, West Lafayette, In., March 1977.

[3] E. Persoon and K. S. Fu, "Sequential Decision Procedures with Prespecified Error Probabilities and Their Applications," TR-EE 74-30, School of Electrical Engineering, Purdue University, West Lafayette, IN, 1974.

[4] S. A. Dudani, et. al., "Aircraft Identification by Moment Invariants," IEEE Trans. on Computers, Vol. C-26, pp. 39-46, Jan. 1977.

A SEMANTIC-SYNTACTIC APPROACH TO IMAGE UNDERSTANDING

G. Y. Tang and T. S. Huang

INTRODUCTION

We propose the injection of semantic features into a context free grammar for the purpose of understanding an input signal.

A feature vector is assigned to each terminal and to each non-terminal. A feature transfer function is attached to each production rule. The feature transfer function transfers features at the right hand side of the production rule to the left hand side non-terminal. After parsing the feature vector associated with the root of the derivation tree is sent to a discriminating function to determine the semantic well-formedness of the sentence.

The acceptance of an input signal is thus based on not only its syntactic structure but also its semantic meaning.

This approach is applicable to many problems in image understanding. In this report we shall present only a simple one-dimensional example.

AN EXAMPLE

We use this approach to find the width of a highway or the location of an edge in aerial photos. The grey level distribution along a straight line segment crossing the highway or edge is obtained by a film scanner.

The apriori knowledge about the signal is that it looks like one of the four paradigms α, β, γ, σ in Fig. 1. α, β are paradigms for ideal edges. γ, σ are the paradigms for highways.

The grammar describing the ideal paradigms is:

1 : 0 → $\hat{D}$ A B ; F1

2 : 0 → $\hat{D}$ B A ; F1

3 : 0 → $\hat{D}$ A ; F7

4 : 0 → $\hat{D}$ B ; F7

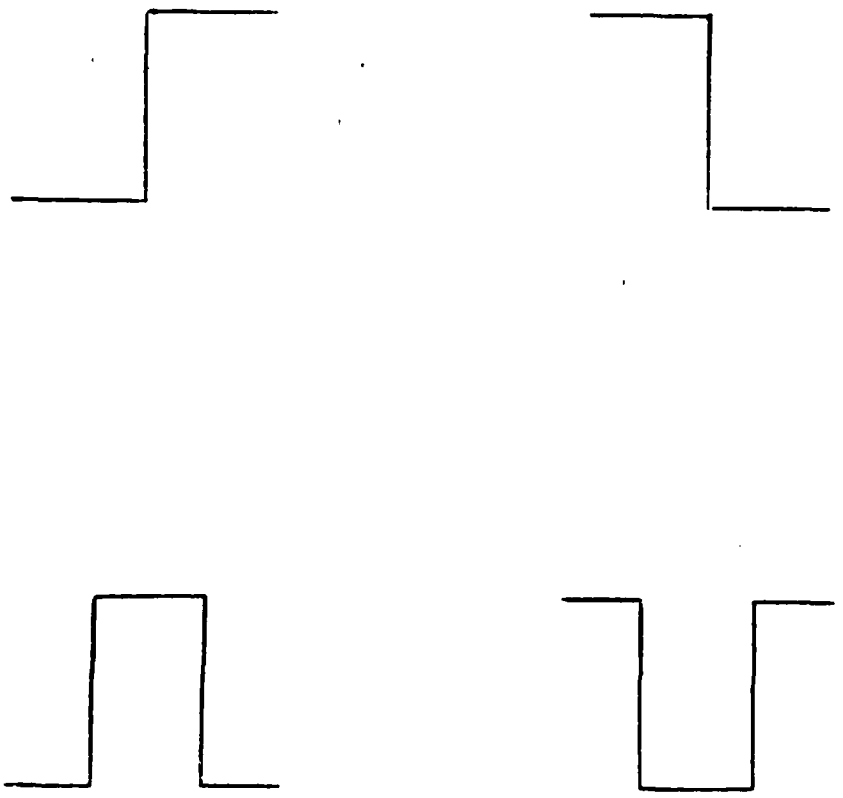


Fig. 1 Three paradigms. α, β, represent edges. γ, σ, represent highways.

5 : A → a X ; F2

6 : X → $\hat{D}$; F3

7 : X → $\hat{D}$ A ; F6

8 : B → b Y ; F2

9 : Y → $\hat{D}$; F3

10 : Y → $\hat{D}$ B ; F6

11 : A → a ; F8

12 : B → b ; F8,

0 is the start symbol, and a, b, $\hat{D}$ are terminals.

The transformation, which brings the ideal paradigms to the realistic level, is to replace each occurrence of the symbol $\hat{D}$ by a sentence generated by the grammar:

T 1 : D → f D ; F4

T 2 : D → c D_1 ; F4

T 3 : D → d D_2 ; F4

T 4 : D → f ; F5

T 5 : D → c ; F5

T 6 : D → d ; F5

T 7 : D_1 → d D_2 ; F4

T 8 : D_1 → d ; F5

T 9 : D_1 → f D ; F4

T10 : D_2 → c D_1 ; F4

T11 : D_2 → c ; F5

T12 : D_2 → f D ; F4

T13 : D_1 → f ; F5

T14 : D_2 → f ; F5,

D is the start symbol, and c, d, f are terminals.

The transfer functions associated with the production rules are defined as:

F1 : $\underline{A} \rightarrow \underline{B}\ \underline{C}\ \underline{D}$

$\tilde{W}(\underline{A}) = \tilde{C}(\underline{D}) - \tilde{C}(\underline{C})$

$\tilde{C}(\underline{A}) = \tilde{C}(\underline{C})$

$\tilde{R2}(\underline{A}) = \begin{cases} 2 & \text{if } \tilde{R2}(\underline{C}) \neq 0 \text{ and } \tilde{R2}(\underline{D}) \neq 0 \\ 0 & \text{otherwise} \end{cases}$

$\tilde{A}(\underline{A}) = |\tilde{A}(\underline{D}) - \tilde{A}(\underline{C})|$

$\tilde{R1}(\underline{A}) = \text{Max}(\text{Max}(\tilde{R1}(\underline{C}), \tilde{R1}(\underline{D})), \tilde{A}(\underline{B})/\tilde{W}(\underline{B}))$

F2 : $\underline{A} \rightarrow \underline{B}\ \underline{C}$

$\tilde{A}(\underline{A}) = \tilde{A}(\underline{B}) + \tilde{A}(\underline{C})$

$\tilde{W}(\underline{A}) = \tilde{W}(\underline{B}) + \tilde{W}(\underline{C})$

$\tilde{C}(\underline{A}) = \begin{cases} \tilde{C}(\underline{B}) & \text{if } \tilde{C}(\underline{C}) = 0 \\ (\tilde{C}(\underline{B}) + \tilde{C}(\underline{C}))/2 & \text{if } \tilde{C}(\underline{C}) \neq 0 \end{cases}$

$\tilde{R1}(\underline{A}) = \tilde{R1}(\underline{B})$

$\tilde{R2}(\underline{A}) = \tilde{R2}(\underline{B})$

F3 : $\underline{A} \rightarrow \underline{B}$

$\tilde{A}(\underline{A}) = \tilde{A}(\underline{B})$

$\tilde{W}(\underline{A}) = \tilde{W}(\underline{B})$

$\tilde{R1}(\underline{A}) = \tilde{A}(\underline{B})/\tilde{W}(\underline{B})$

$\tilde{R2}(\underline{A}) = \tilde{W}(\underline{B})$

$\tilde{C}(\underline{A}) = 0$

F4 : $\underline{A} \rightarrow \underline{B}\ \underline{C}$

$\tilde{A}(\underline{A}) = \text{Max}(\tilde{A}(\underline{B}), \tilde{A}(\underline{C}))$

$\tilde{W}(\underline{A}) = \tilde{W}(\underline{B}) + \tilde{W}(\underline{C})$

$\tilde{C}(\underline{A}) = 0$

F5 : $\underline{A} \rightarrow \underline{B}$

$\tilde{C}(\underline{A}) = 0$

$\tilde{A}(\underline{A}) = \tilde{A}(\underline{B})$

$\tilde{W}(\underline{A}) = \tilde{W}(\underline{B})$

F6 : $\underline{A} \rightarrow \underline{B}\ \underline{C}$

$\tilde{A}\ (\underline{A}) = \tilde{A}\ (\underline{C})$

$\tilde{W}\ (\underline{A}) = \tilde{W}\ (\underline{C})$

$\tilde{R1}\ (\underline{A}) = \tilde{R1}\ (\underline{C})$

$$\tilde{R2}\ (\underline{A}) = \begin{cases} 0 \text{ if } \tilde{W}\ (\underline{B}) \geq t \\ \tilde{R2}\ (\underline{C}), \text{ otherwise} \end{cases}$$

$\tilde{C}\ (\underline{A}) = \tilde{C}\ (\underline{C})$

F7 : $\underline{A} \rightarrow \underline{B}\ \underline{C}$

$\tilde{R1}\ (\underline{A}) = \text{Max}\ (\tilde{R1}\ (\underline{C}), \tilde{A}\ (\underline{B})/\tilde{W}\ (\underline{B}))$

$\tilde{C}\ (\underline{A}) = \tilde{C}\ (\underline{C})$

$\tilde{W}\ (\underline{A}) = \tilde{W}\ (\underline{C})$

$$\tilde{R2}\ (\underline{A}) = \begin{cases} 1 \text{ if } \tilde{R2}\ (\underline{C}) \neq 0 \\ 0 \text{ otherwise} \end{cases}$$

F8 : $\underline{A} \rightarrow \underline{B}$

$\tilde{C}\ (\underline{A}) = \tilde{C}\ (\underline{B})$

$\tilde{W}\ (\underline{A}) = \tilde{W}\ (\underline{B})$

$\tilde{A}\ (\underline{A}) = \tilde{A}\ (\underline{B})$

$\tilde{R1}\ (\underline{A}) = 1.$

The underscored symbols like $\underline{A}$ are formal parameters. Symbols with ~ atop designate features associated with the formal parameters in parentheses, e.g. $\tilde{C}\ (\underline{A})$ is the $\tilde{C}$ feature attached to $\underline{A}$.

There are five terminals a, b, c, d, f. To each terminal, there are three features attached. Literally a, b, c, d, and f are five tendencies in the input signal. a and c represent the tendency of going-up. b and d are for going-down. f is for flatness. The extent of going-up differentiates a from c. a stands for long going-up. c stands for short going-up. Similarily b is long going-down and d is short going-down.

The three features attached to the terminals are $\tilde{A}$, $\tilde{W}$, and $\tilde{C}$. $\tilde{C}$ refers to the center of the tendency. $\tilde{W}$ refers to the width of the tendency. $\tilde{A}$ is a measure of the opposition (long/short). $\tilde{A}$ (a) = 1 or $\tilde{A}$ (b) = 1 means absolutely long. $\tilde{A}$ (c) = 1 or $\tilde{A}$ (d) = 1 means absolutely short. More specifically, let $\ell(\cdot)$ denote the height of the tendency. Then for a, b, we have

$$\tilde{A}(S) = (\ell(S)/M_1 - t)/(1-t).$$

For c, d, we have

$$\tilde{A}(S) = [\ell(S)/M_1 - t)/t] + 1.$$

M_1 is the maximum height. $t\,M_1$ is the threshold for discriminating between "long" and "short".

For non-terminals, there are two more features. These two features are defined by the transfer functions.

The final semantic well-formedness test is:

$\tilde{W}(0) \geq t_1$, $\tilde{A}(0) \leq t_3$

$\tilde{R}1(0) \geq t_2$

$\tilde{R}2(0) > 0.$

$\tilde{C}(0)$ is the location of the edge or the front edge of the highway. $\tilde{W}(0)$ is the width. $\tilde{R}2(0) = 1$ indicates edges. $\tilde{R}2(0) = 2$ indicates highway. t_1, t_2, and t_3 are preset thresholds.

EXPERIMENTAL RESULTS

The experimental results are shown in Figs. 2-7.

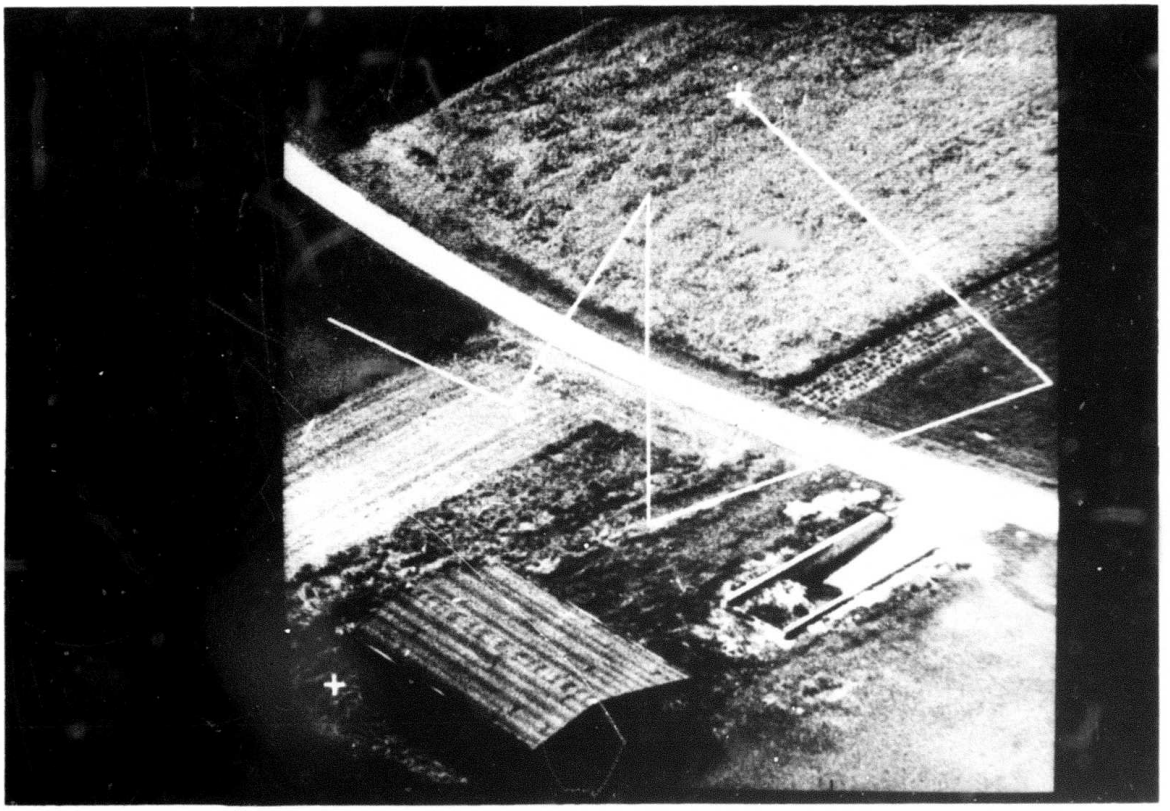


Fig. 2 Original picture. The grey level distribution along the straight line are the input signals. Fig. 3 is the grey level distribution of the left-most line segment. Fig. 4 is the one next to Fig. 3. Fig. 5 is the one next to Fig. 4. Fig. 6 is next to Fig. 5. Fig. 7 is next to Fig. 6. From Fig. 3 to Fig. 7, we look for highway only.

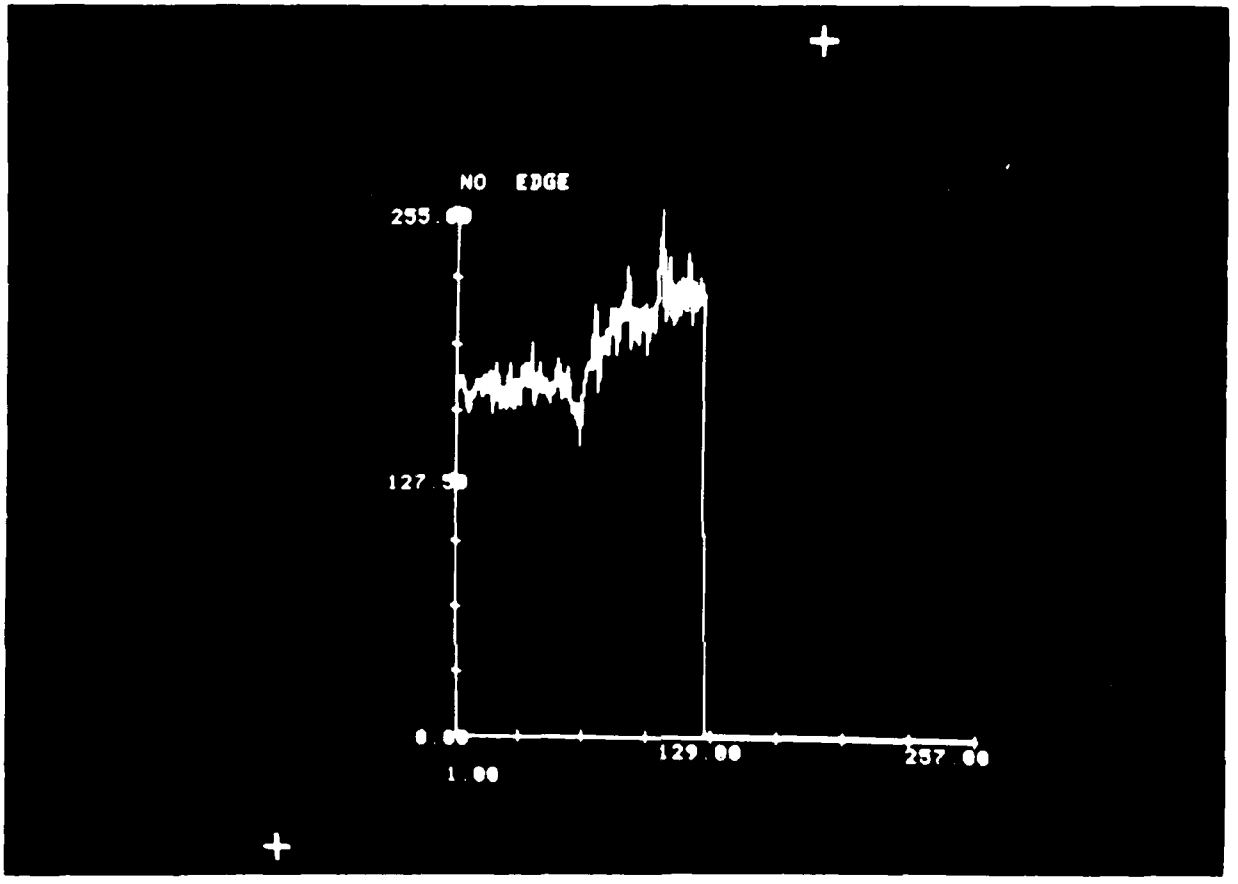


Fig. 3 No highway is found

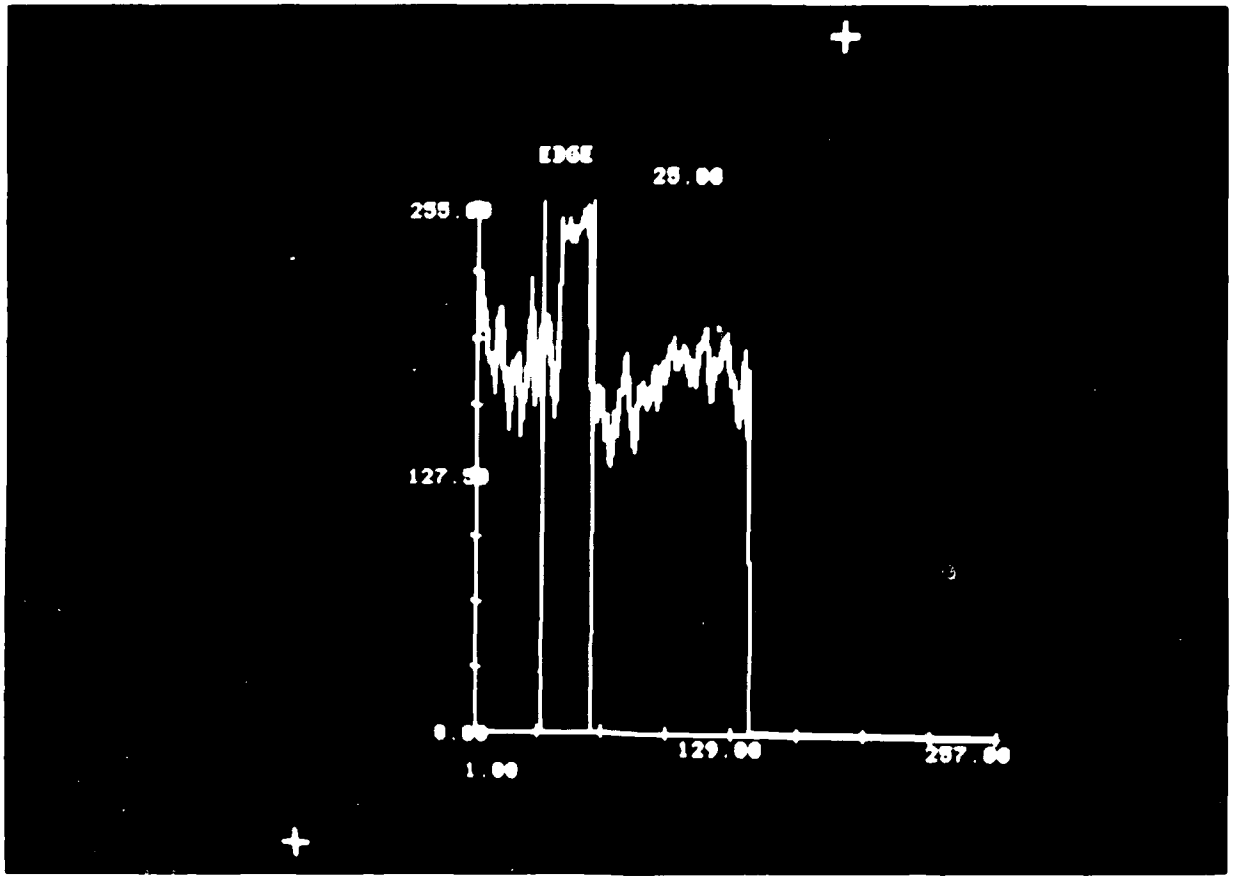


Fig. 4 Highway is defined by two vertical lines. The width of highway is 25 pixels.

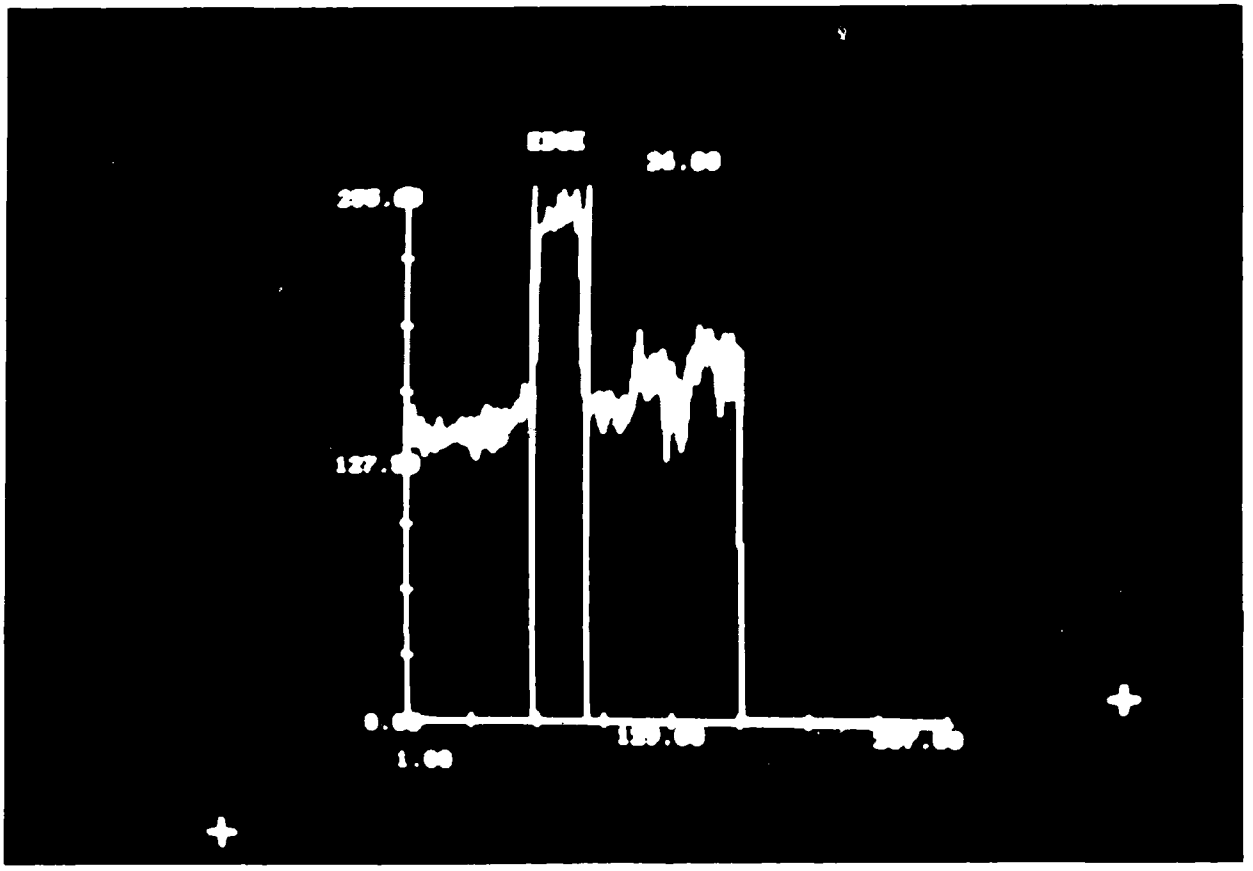


Fig. 5 Highway is defined by two vertical lines.
Its width is 26 pixels.

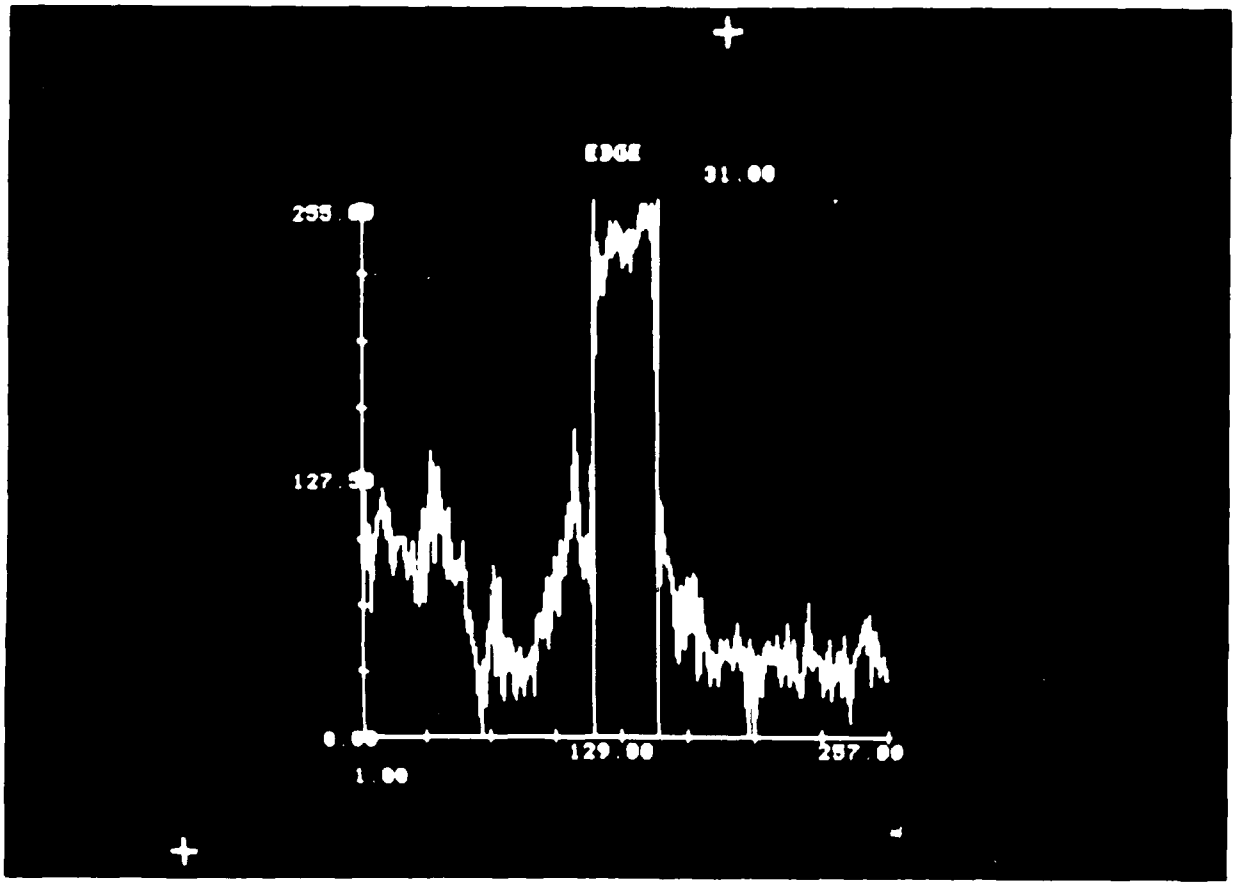


Fig. 6 Highway is defined by two vertical lines. Its width is 31 pixels.

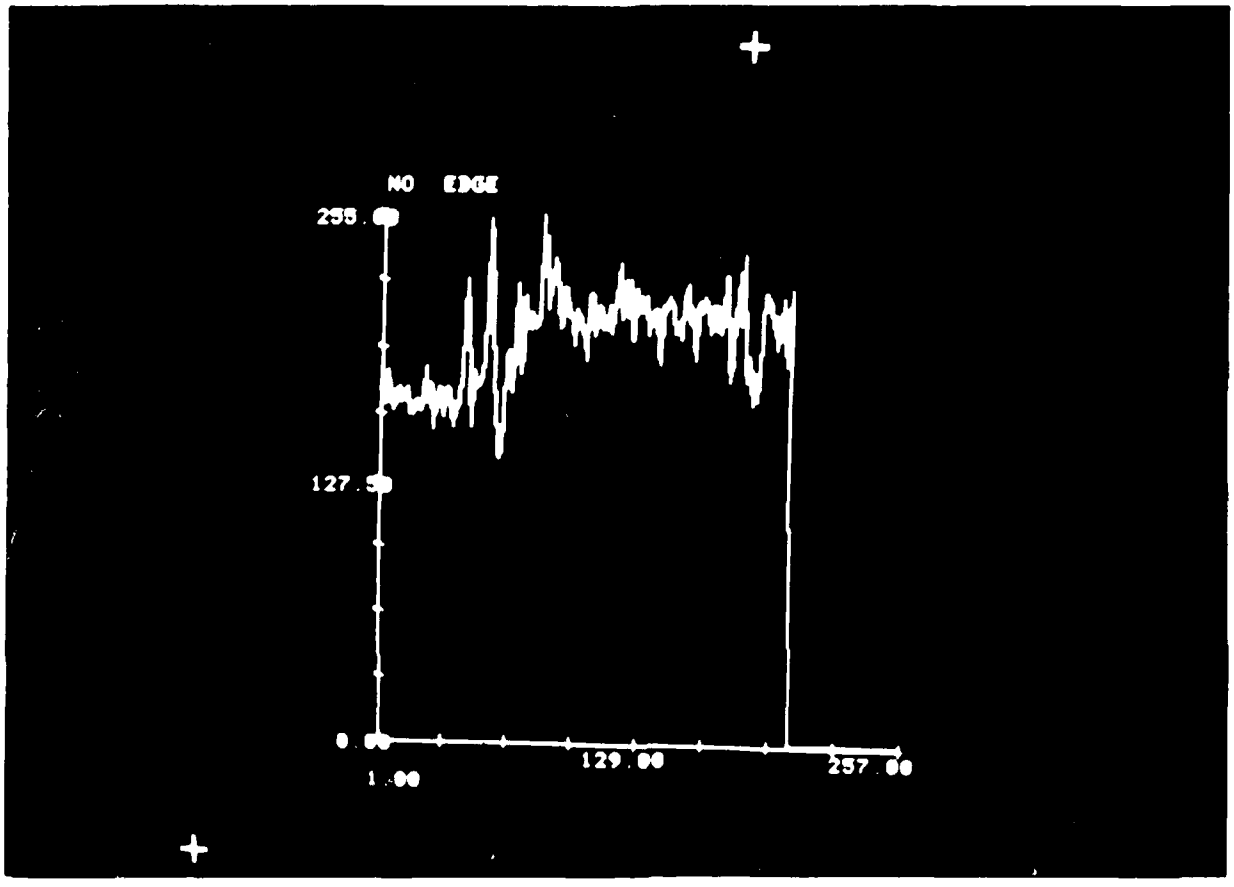


Fig. 7 No highway is found

SYNTACTIC SHAPE RECOGNITION

K. S. Fu and K. C. You

INTRODUCTION

Since the shape of an object , defined as its outer boundary, provides important information from its structure and local boundary details, the syntactic method [1] is proposed to fully utilize this information for recognition purpose. In the last report [2], the primitive descriptors and the shape grammar have been discussed. In the following sections of this report, the normalization of the descriptors, the implementation of a bottom-up parsing scheme and a recognition experiment are described.

NORMALIZATION OF PRIMITIVE DESCRIPTORS

To recognize a primitive in the boundary, we simply rely on the similarity measurement between the primitive descriptors. But the digitization introduces different noise to the descriptor with respect to any operation of rotation, scaling and translation. Normally, the digitization noise introduced due to rotation is much more significant than that due to scaling and translation, because the rotation influences the angle feature significantly. Although we can apply some smoothing techniques to the boundary vector chain, the true boundary can not be recovered in all cases completely. We need to study the possible distribution of the feature values under various rotations to help constructing a proper similarity measurement.

In following paragraphs, we would concentrate on the influence of rotation to the shape of information of a curve primitive. The shape information is characterized by $(^{C}/L, A, ^{S}/L)$, which can be obtained from the descriptor $(\vec{C}, L, A, S)$ [2].

The normalization function is defined as follows:

Definition 1:

Normalization function, N, of the curve primitive descriptor is:

$N: (\vec{C}, L, A, S) \rightarrow (X, Y, Z)$

where $X = C/L \quad 0 \leq X \leq 1$

$Y = A/2\pi$ (angle in terms of revolution)

$Z = S/AL$, $-0.5 < Z < 0.5$ for a simple curve segment [2]

An experiment was designed and carried out through following steps to study the distribution of the normalized variables under different rotations.

Experiment:

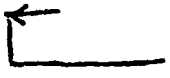
1. A picture with clear boundary was scanned with respect to 8 various rotation angles.
2. Shapes on the digital pictures were traced out and passed through a smoothing procedure. The vector chains were obtained.
3. Manual extraction of primitives from the chain was performed via an interactive procedure.
4. The descriptors of the manually extracted primitives were computed and transformed by N.
5. Studied the distributions of the normalized variables.

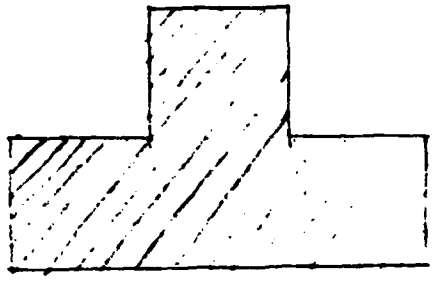
The three normalized variables, X, Y, Z, constitute a three-dimensional space, named 3-D for short. Table 2.1 illustrates the pictures used, the curve primitives and the corresponding symbols in following figures. Figure 2.1 - 2.3 show the two-dimensional displays of the distributions of descriptors, each symbol in the figure indicates the position of a descriptor in 3-D.

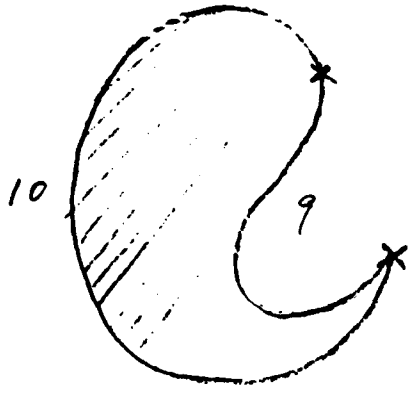
Several interesting aspects are observed:

(1) The 3 variables well characterize the shape. Any pair of clusters can be separated in at least one of the 2-dim. displays.

Table 2.1

Cluster	Symbol	Curve Prim.
1	○	
2	△	
3	+	
4	×	
5	◇	
6	⏃	
7	⧖	
8	Z	
9	Y	
10	⊠	





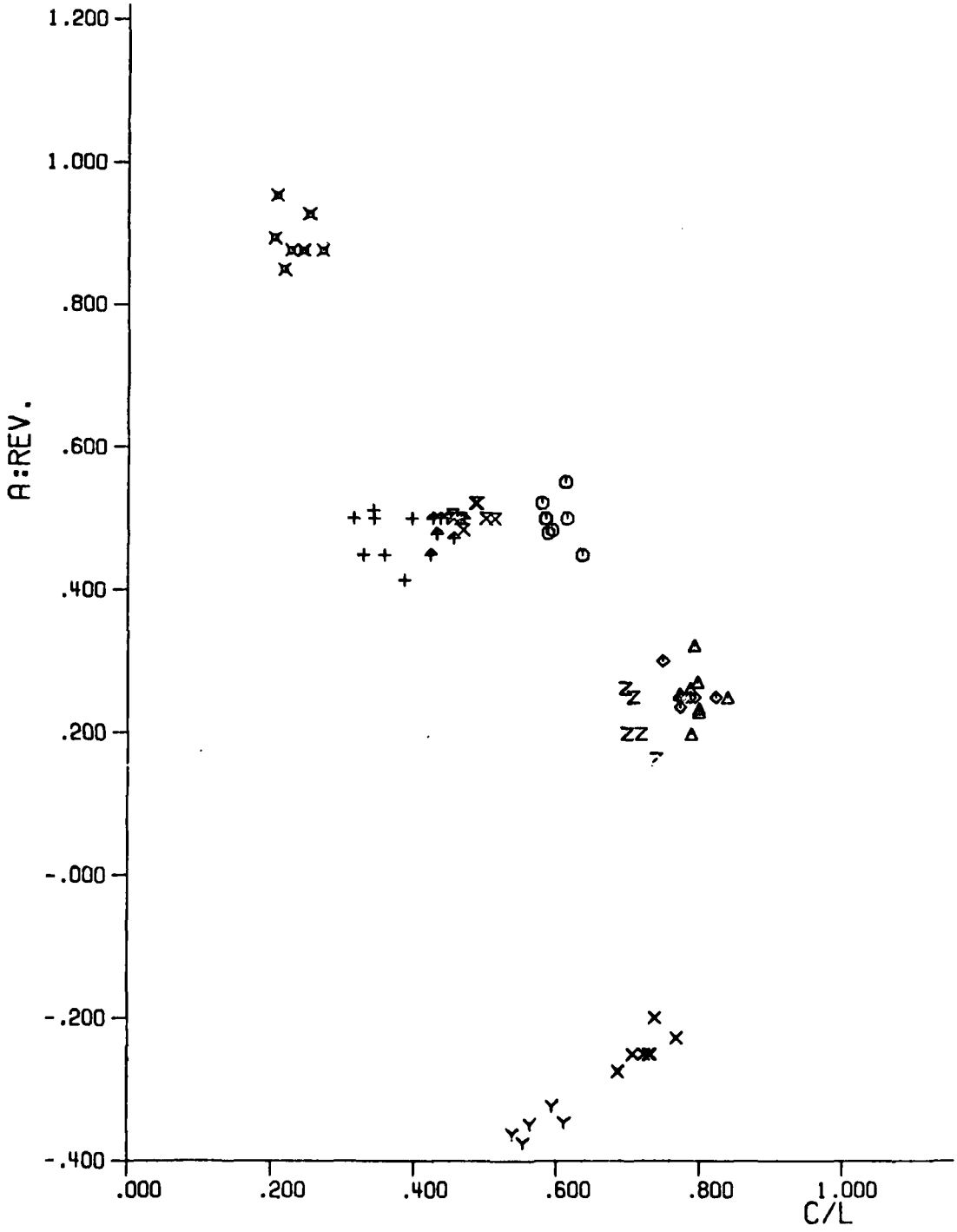

1.200
1.000
.800
.600
.400
.200
-.000
-.200
-.400
A:REV.
.000
.200
.400
.600
.800
1.000
C/L

Fig. 2.1

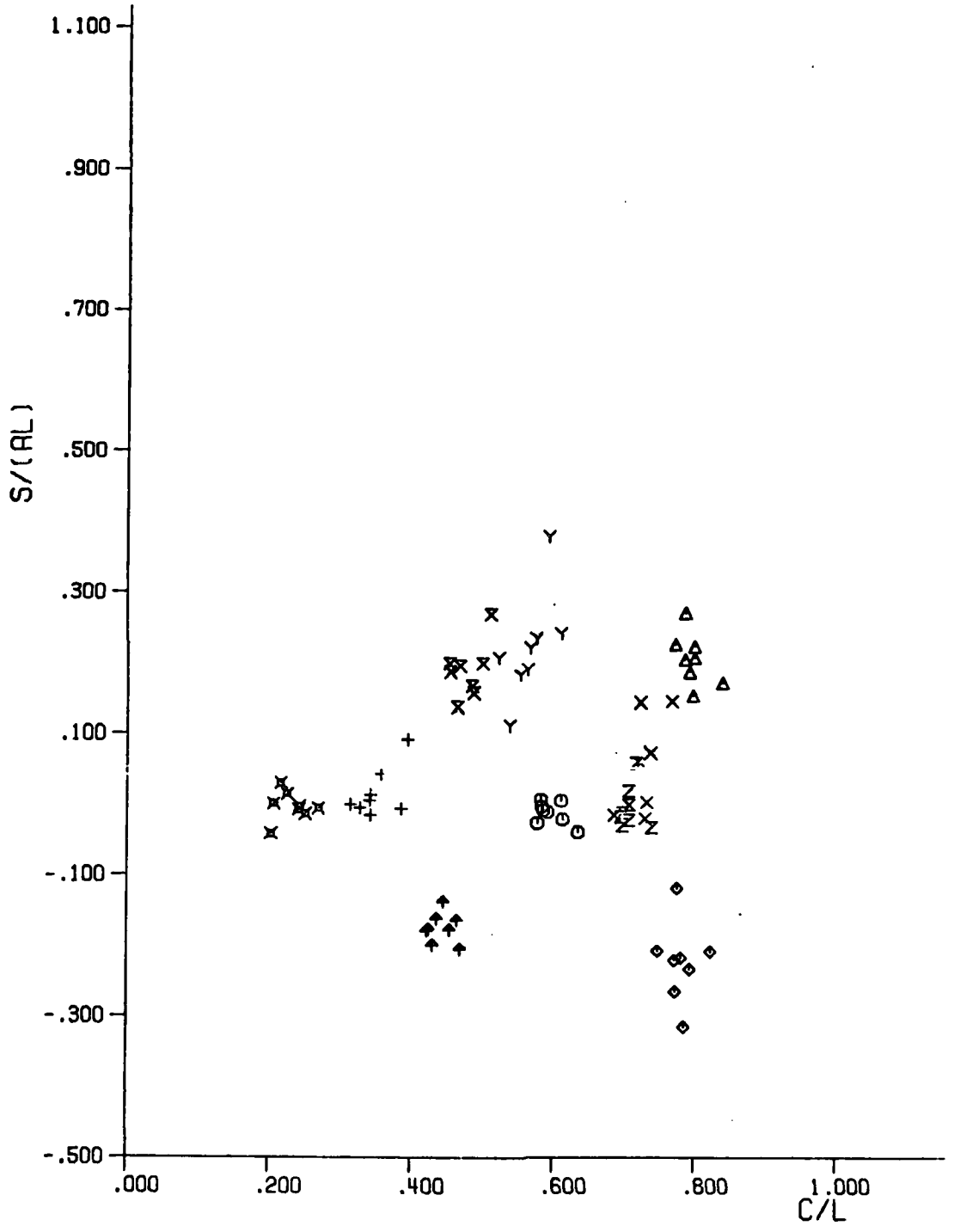

1.100
.900
.700
.500
.300
.100
-.100
-.300
-.500
S/(AL)
.000
.200
.400
.600
.800
1.000
C/L

Fig. 2.2

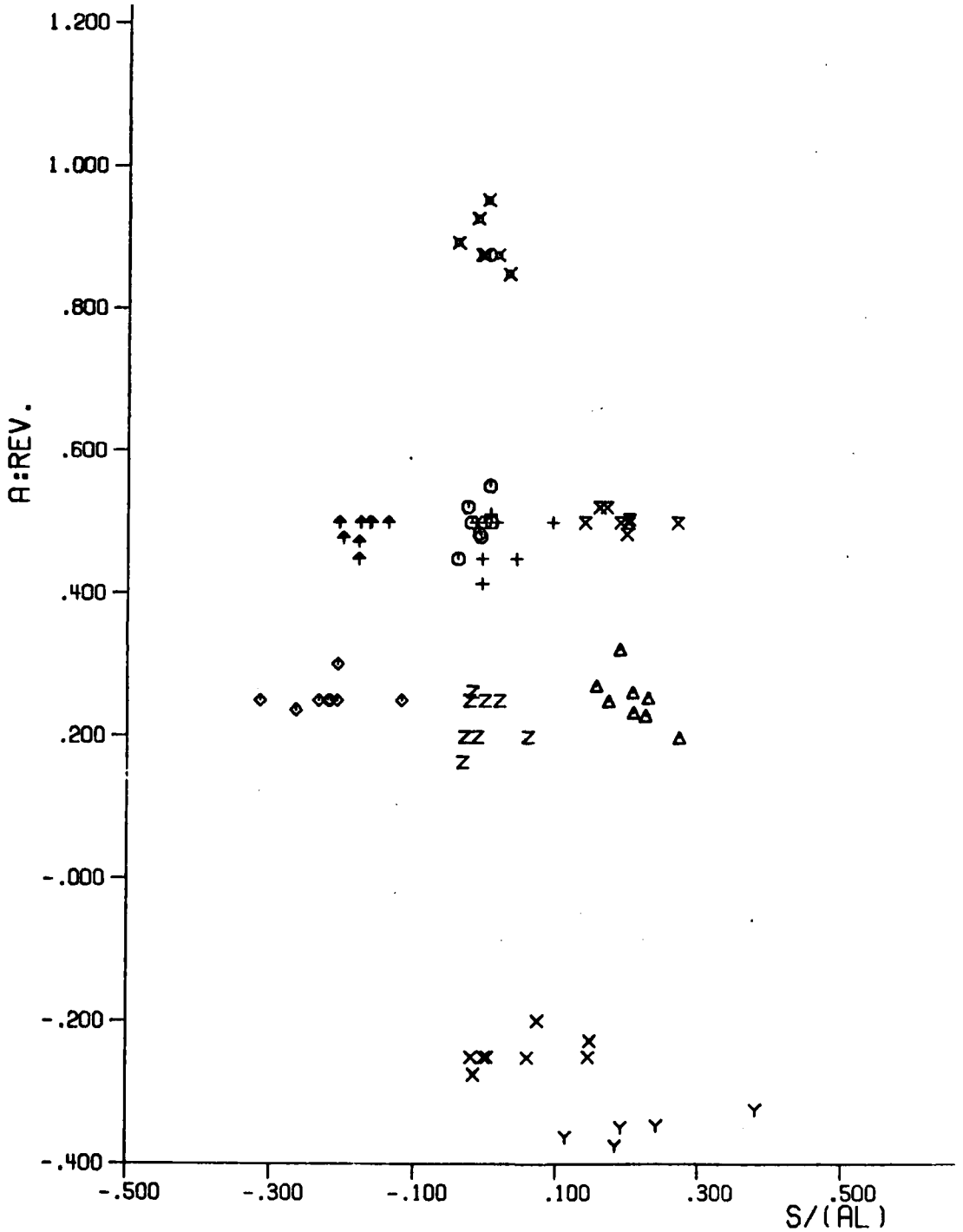

1.200
1.000
.800
.600
.400
.200
-.000
-.200
-.400
A:REV.
-.500
-.300
-.100
.100
.300
.500
S/(AL)

Fig. 2.3

(2) The number of points is not enough to reveal a parametric distribution, but the points within each cluster are considerably close together.

(3) The variable $Z = \frac{S}{AL}$ is more spread-out than the other two. The reason might be that the noises in A and L are accumulated in the calculation of S, which is the summation of the partial products of A and L.

This experiment is not sufficient to demonstrate the distribution. Besides, the distribution could be changed with the boundary smoothing techniques. However, this experiment gives us an idea to construct the similarity measurement.

For the simplicity of computation, we may assume that each curve primitive has a reference point, or prototype point, in the 3-D. The descriptors of a subchain of vectors are compared with the reference points via a distance measurement. This distance measurement between two shapes can be defined as $D_s(q_1,q_2)$.

Definition 2:

$D_s(q_1,q_2)$ is the distance between two curves q_1 and q_2, where $N(q_i) = (X_i, Y_i, Z_i)$, and $D_s(q_1,q_2) = \text{func}\ (X_1, X_2, Y_1, Y_2, Z_1, Z_2)$.

As mentioned in [2], the relative size of a curve segment to the whole boundary also provides information for identifying the primitives. We define another measurement for the size difference.

Definition 3:

$S_d(q_1,q_2)$ measures the difference between the relative sizes of two curves q_1 and q_2.

$T(q_i) = (^{C}i/L_i, {}^{L}i/L_{10}, A_i, {}^{S}i/L_i)$ [2]

$S_d(q_1,q_2) = \text{func}\ ({}^{L}1/L_{10}, {}^{L}2/L_{20})$

IMPLEMENTATION OF A RECOGNIZER

A conventional syntactic recognizer contains two steps: primitive extraction and parsing. The former step converts the unknown pattern into a presentation of primitives and the later step checks the structure of the presentation with the grammar rules. If the structure fits the rules, the unknown pattern is accepted, otherwise rejected.

Our recognizer combines the two steps into one, the advantages would be explained later. The Earley's parsing algorithm [3] has been modified to achieve our task. For recognition purpose, we need only the first part of Earley's algorithm, i.e., the generation of parsing table. The algorithm is modified so that it accepts the boundary vector chain instead of the primitive string as its input. In other words, the primitives extraction is embedded in the parsing table generation.

The context-free shape grammar G is of the form:

$G = (V_N, V_T, P, S)$ [2]

where

V_T = {S, N's|N: nonterminals}

V_T = {F's, A's|F: curve primitive, A: angle primitive}

Before discussing the algorithm, some definitions are given as follows:

Definition 4:

v_1 is a vector

v_1 - - v_n is a closed vector chain.

D(X) is the descriptor of a terminal/nonterminal

$X \varepsilon ((V_N \cup V_T) - S)$

D(i,j) denotes the descriptor of the subchain v_i - - v_j,

$D(X) = D(i,j)$ implies that subchain v_i -- v_j can be recognized as X

$I_1, I_2, \ldots, I_{n+1}$ are the parse lists

For item $[A \to \alpha \cdot \beta, i]$ in I_j, $1 \leq i \leq j$, means

(1) iff $\alpha \neq \lambda$, (empty string) then $\alpha \overset{*}{\Rightarrow} v_i v_{i+1} - - v_j$

(2) if $\alpha = \lambda$, then $i = j$

The following assumption makes it possible to recognize a primitive.

Assumption:

$D(X) \simeq D(i,j)$, iff $D_s(N(X), N(i,j)) < d$, and

$S_d(X,(i,j)) < s$, where d, s are thresholds.

The Modified Algorithm

(1) Add $[S \to \cdot \alpha, 1]$ to I_1

for all $S \to \alpha$ in P

$j = 1$

(2) (a) if $[N \to \alpha \cdot B\beta, i]$ is in I_j

$B \to \gamma$ in P

add $[B \to \cdot \gamma, j]$ to I_j.

(b) $[N \to \alpha \cdot, i]$ is in I_j

then for all $[B \to \beta \cdot N\gamma, k]$ in I_i

add $[B \to \beta N \cdot \gamma, k]$ to I_j

(3) $j = j+1$

if $j > n+1$ go to (4)

For all $[N \to \alpha \cdot X\beta, i]$ in I_k, $1 \leq k \leq j$ $X \in \{F's, A's\}$

(a) if $\beta \neq \lambda$ and $D(X) \simeq D(k,j)$

then add $[N \to \alpha X \cdot \beta, i]$ to I_j

(b) if $\beta = \lambda$, $D(X) \simeq D(k,j)$ and $D(N) \simeq D(i,j)$

then add $[N \to \alpha X \cdot , i]$ to I_j

go to (2)

(4) if $[S \to \alpha \cdot , 1]$ in I_{n+1} for some α, then $w \in L(G)$

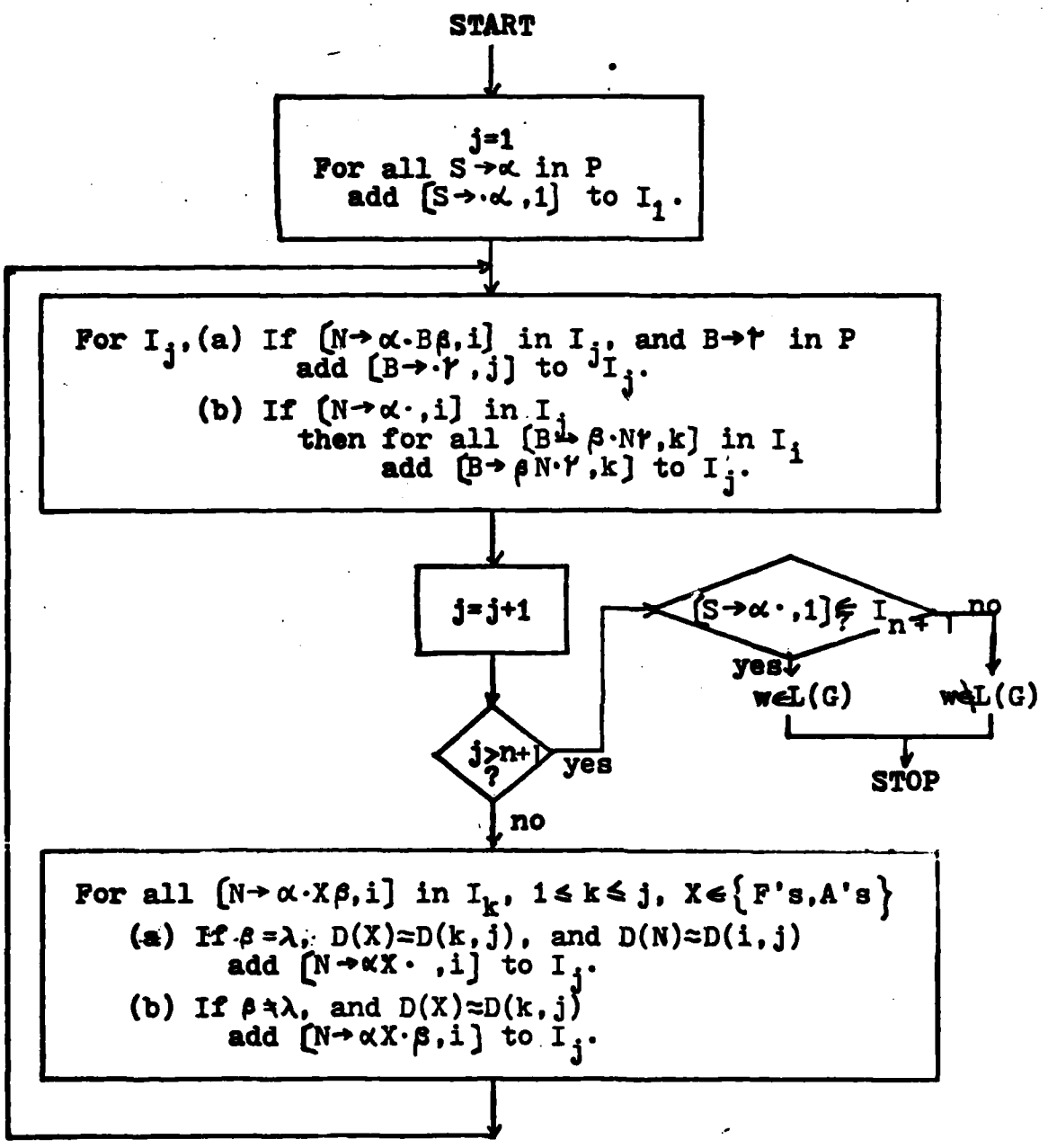

START
j=1
For all S→α in P
add [S→·α,1] to I1.
For Ij, (a) If [N→α·Bβ,i] in Ij, and B→γ in P
add [B→·γ,j] to Ij.
(b) If [N→α·,i] in Ij
then for all [B→β·Nγ,k] in Ij
add [B→βN·γ,k] to Ij.
j=j+1
[S→α·,1] ∈? In+1
no
yes
w∈L(G)
w∉L(G)
STOP
j>n+1 ?
yes
no
For all [N→α·Xβ,i] in Ik, 1≤k≤j, X∈{F's,A's}
(a) If β=λ, D(X)≈D(k,j), and D(N)≈D(i,j)
add [N→αX·,i] to Ij.
(b) If β≠λ, and D(X)≈D(k,j)
add [N→αX·β,i] to Ij.

Fig. 3.1

The following example illustrates why we combine the primitive extraction into the parsing scheme. Let us see the step (3.a). In extracting F_1 of $[N \rightarrow \alpha \cdot F_1, i_0]$ in I_{K_0} from the vector chain $D(F_1) \approx D(K_0,j)$ and $D(N) \approx D(i_0,j)$ may be valid for both $j = j_1$ and j_2. In other words, subchains $v_{K_0} - - v_{j_1}$ and $v_{K_0} - - v_{j_2}$ are candidates for F_1, so $[N \rightarrow \alpha F_1 \cdot i_0]$ is in I_{j_1} and $[N \rightarrow \alpha F_1 \cdot , i_0]$ is in I_{j_2}. Suppose that $j_1 < j_2$. After the execution of step (2) for $j = j_2$, $[B \rightarrow \beta N \cdot \gamma, K_1]$ is in I_{j_1} and $[B \rightarrow \beta N \cdot \gamma, K_1]$ is in I_{j_2} where $K_1 \leq i_0 \leq K_0 \leq j_1 < j_2$. Suppose that $\gamma = A_1F_2A_2F_3$, the execution of step (3.b) to extract A_1 of $[B \rightarrow \beta N \cdot A_1F_2A_2F_3, K_1]$ in I_{j_1} and I_{j_2} from the vector chain may find out that $D(A_1) \not\approx D(j_1,j_3)$ for any $j_3 > j_1$, but $D(A_1) \approx D(j_2,j_4)$ for some $j_4 > j_2$, then only $[B \rightarrow \beta N A_1 \cdot F_2A_2F_3, K_1]$ is added to I_{j_4}. That is, the context information is used to select the subchain $v_{K_0} - - v_{j_2}$ for F_1 and discard $v_{K_0} - - v_{j_1}$. If $D(A_1) \approx D(j_1,j_3)$ for some $j_3 > j_1$, and $D(A_1) \approx D(j_2,j_4)$ for some $j_4 > j_2$, then $[B \rightarrow \beta N A_1 \cdot F_2A_2F_3, K_1]$ is in I_{j_3} and $[B \rightarrow \beta N A_1 \cdot F_2A_2F_3, K_1]$ is in I_{j_4}. The execution of step (3.b) to extract F_2 from the vector chain may find out that $D(F_2) \not\approx D(j_3,j_5)$ for any $j_5 \geq j_3 > j_2$, but $D(F_2) \approx D(j_4,j_6)$ for some $j_6 \geq j_4 > j_1$, then only $[B \rightarrow \beta N A_1F_2 \cdot A_2F_3, K_1]$ is added to I_{j_6}. That is, the lookahead on the information of subchain $v_{j_1} - - v_{j_5}$ selects the candidate $v_{K_0} - - v_{j_2}$ for F_1.

In fact, the extraction of A's and F's embedded in our parsing is different from the pre-extraction of the primitives without knowing the context information. The advantage is that the extraction would be much more accurate in a global sense.

EXPERIMENTAL RESULTS: SHAPE RECOGNITION OF THE TOP VIEWS OF AIRPLANES

The main purpose of this experiment is to testify whether the proposed method is capable of recognizing shape. The idea is to recognize noisy shapes

from digital images by using the modified Earley's parsing algorithm with given models and shape grammars.

In constructing shape grammars, we must consider how to obtain a better segmentation of a shape. Since the picture is scanned line by line, the convex points of the object are usually the first point hit in the scanning, and hence, the starting point of the vector chain. Therefore, our grammars and breaking points, which break the shape into small curve segments, are designed such that most of the possible sentences with respect to different starting points are contained in the generated languages.

If a noisy picture is unfortunately distorted at a certain breaking point, the descriptors of the two related curve segments may be badly affected, because the direction of the two ends of a curve segment affects the feature values of the descriptor. In such cases, the machine may misrecognize. The solution is to utilize more than one set of segmentation. If a breaking point with respect to one segmentation is distorted badly, it can be passed around by another set of segmentation, because it may not be the breaking point in that segmentation. The noise occured not at the ends has little effect. In this experiment, each of the grammars utilizes essentially two sets of segmentation.

A less noisy shape may be recognized by more than one set of segmentation. In other words, there may be more than one derivation for a sentence. This intended ambiguity [1] increases the recognition power for noisy shapes.

Nine noisy shapes have been tried against three grammars, three models of airplanes, with the modified Earley's parser. All of the 27 tests were correctly accepted or rejected.

The following pages are the shape grammars, and their corresponding segmentation graph. The nine noisy shapes are shown in Figure 4.7 - 4.15. The small circle at the bottom of each shape indicates the starting point of the boundary vector chain.

REFERENCES

[1] K. S. Fu, Syntactic Method in Pattern Recognition, Academic Press, 1974.

[2] K. C. You and K. S. Fu, "Syntactic Shape Recognition," ARPA Quarterly Report,

[3] A. V. Aho and J. D. Ullman, Theory of Parsing Translation and Compiling, Vol. 1, Parsing, Prentice-Hall, 1972.

The shape grammar for AIRBUS, $G_b = (V_b, T_b, P_b, S_b)$

$V_b = \{S_b, N_{bi} \mid 1 \leq i \leq 5\}$

$T_b = \{F_{bj}, A_{bk} \mid 1 \leq j \leq 22, 1 \leq k \leq 10\}$

P_b:

(1) $S_b \rightarrow N_{b1}A_{b1}F_{b1}A_{b2}N_{b2}A_{b3}N_{b3}A_{b3}N_{b4}A_{b2}F_{b2}A_{b1}N_{b5}A_{b4}$

(2) $S_b \rightarrow F_{b1}A_{b2}N_{b2}A_{b3}N_{b3}A_{b3}N_{b4}A_{b2}F_{b2}A_{b1}N_{b5}A_{b4}N_{b1}A_{b1}$

(3) $S_b \rightarrow N_{b2}A_{b3}N_{b3}A_{b3}N_{b4}A_{b2}F_{b2}A_{b1}N_{b5}A_{b4}N_{b1}A_{b1}F_{b1}A_{b2}$

(4) $S_b \rightarrow N_{b3}A_{b3}N_{b4}A_{b2}F_{b2}A_{b1}N_{b5}A_{b4}N_{b1}A_{b1}F_{b1}A_{b2}N_{b2}A_{b3}$

(5) $S_b \rightarrow N_{b4}A_{b2}F_{b2}A_{b1}N_{b5}A_{b4}N_{b1}A_{b1}F_{b1}A_{b2}N_{b2}A_{b3}N_{b3}A_{b3}$

(6) $S_b \rightarrow F_{b2}A_{b1}N_{b5}A_{b4}N_{b1}A_{b1}F_{b1}A_{b2}N_{b2}A_{b3}N_{b3}A_{b3}N_{b4}A_{b2}$

(7) $S_b \rightarrow N_{b5}A_{b4}N_{b1}A_{b1}F_{b1}A_{b2}N_{b2}A_{b3}N_{b3}A_{b3}N_{b4}A_{b2}F_{b2}A_{b1}$

(8) $N_{b1} \rightarrow F_{b3}A_{b5}F_{b4}A_{b6}F_{b5}A_{b7}F_{b6}$

(9) $N_{b1} \rightarrow F_{b7}A_{b8}F_{b8}A_{b7}F_{b6}$

(10) $N_{b2} \rightarrow F_{b9}A_{b9}F_{b10}A_{b10}F_{b11}$

(11) $N_{b4} \rightarrow F_{b12}A_{b10}F_{b13}A_{b9}F_{b14}$

(12) $N_{b3} \rightarrow F_{b15}A_{b11}F_{b16}$

(13) $N_{b5} \rightarrow F_{b17}A_{b7}F_{b18}A_{b6}F_{b19}A_{b5}F_{b20}$

(14) $N_{b5} \rightarrow F_{b17}A_{b7}F_{b21}A_{b8}F_{b22}$

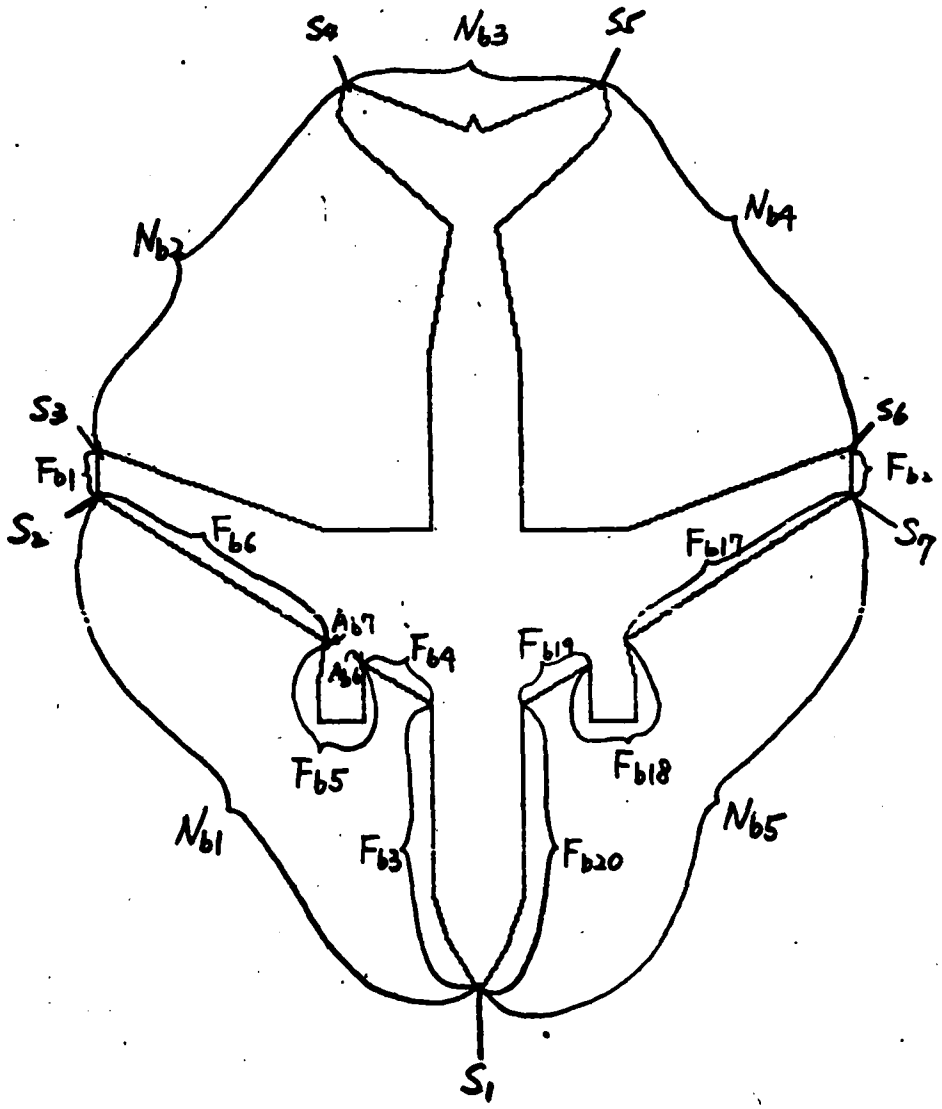

S4
Nb3
S5
Nb2
Nb4
S3
S6
Fb1
Fb2
S2
S7
Fb6
Fb17
Fb7
Fb9
Fb19
Fb5
Fb18
Nb1
Nb5
Fb3
Fb20
S1

Fig. 4.1

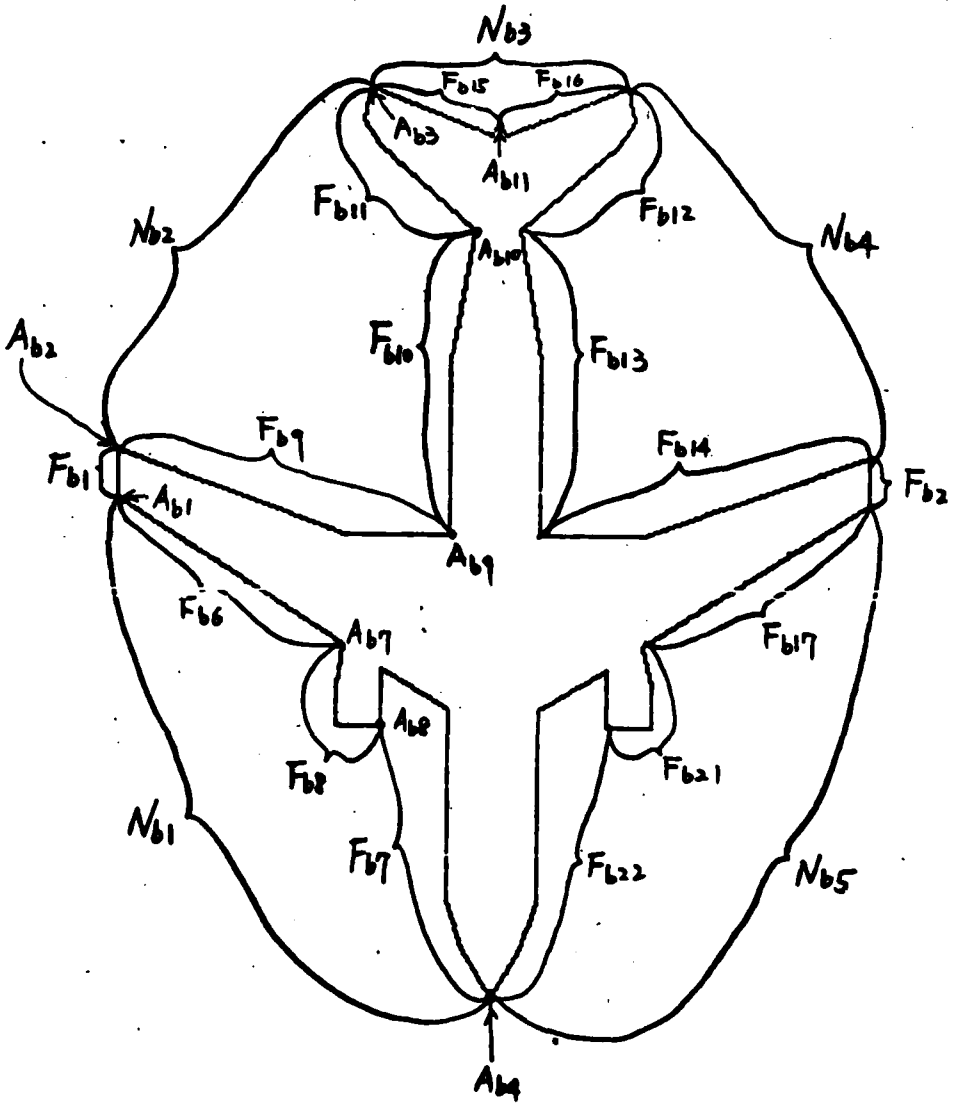

N_{b3}
F_{b15}
F_{b16}
A_{b3}
A_{b11}
F_{b11}
F_{b12}
N_{b2}
A_{b10}
N_{b4}
A_{b2}
F_{b10}
F_{b13}
F_{b9}
F_{b14}
F_{b1}
F_{b2}
A_{b1}
A_{b9}
F_{b6}
A_{b7}
F_{b17}
A_{b8}
F_{b8}
F_{b21}
N_{b1}
F_{b7}
F_{b22}
N_{b5}
A_{b4}

Fig. 4.2.

The shape grammar for DC8, $G_d = (V_d, T_d, P_d, S_d)$.

$V_d = \{ S_d, N_{di} \mid 1 \leq i \leq 15 \}$

$T_d = \{ F_{dj}, A_{dk} \mid 1 \leq j \leq 27, 1 \leq k \leq 12 \}$

P_d:

(1) $S_d \rightarrow N_{d1}A_{d1}N_{d2}A_{d2}F_{d1}A_{d2}N_{d3}A_{d1}N_{d4}A_{d3}$

(2) $S_d \rightarrow N_{d2}A_{d2}F_{d1}A_{d2}N_{d3}A_{d1}N_{d4}A_{d3}N_{d1}A_{d1}$

(3) $S_d \rightarrow F_{d1}A_{d2}N_{d3}A_{d1}N_{d4}A_{d3}N_{d1}A_{d1}N_{d2}A_{d2}$

(4) $S_d \rightarrow N_{d3}A_{d1}N_{d4}A_{d3}N_{d1}A_{d1}N_{d2}A_{d2}F_{d1}A_{d2}$

(5) $S_d \rightarrow N_{d4}A_{d3}N_{d1}A_{d1}N_{d2}A_{d2}F_{d1}A_{d2}N_{d3}A_{d1}$

(6) $S_d \rightarrow N_{d6}A_{d2}F_{d1}A_{d2}N_{d7}A_{d4}N_{d8}A_{d3}N_{d5}A_{d4}$

(7) $S_d \rightarrow N_{d8}A_{d3}N_{d5}A_{d4}N_{d6}A_{d2}F_{d1}A_{d2}N_{d7}A_{d4}$

(8) $N_{d1} \rightarrow N_{d9}A_{d5}N_{d10}$

(9) $N_{d5} \rightarrow N_{d9}A_{d5}N_{d11}$

(10) $N_{d9} \rightarrow N_{d12}A_{d6}F_{d2}$

(11) $N_{d12} \rightarrow F_{d3}A_{d7}F_{d4}$

(12) $N_{d12} \rightarrow F_{d5}A_{d8}F_{d6}A_{d9}F_{d7}$

(13) $N_{d10} \rightarrow F_{d8}A_{d10}F_{d9}$

(14) $N_{d11} \rightarrow F_{d10}A_{d11}F_{d11}$

(15) $N_{d2} \rightarrow F_{d12}A_{d12}F_{d13}$

(16) $N_{d6} \rightarrow F_{d14} A_{d12} F_{d13}$

(17) $N_{d3} \rightarrow F_{d15} A_{d12} F_{d16}$

(18) $N_{d7} \rightarrow F_{d15} A_{d12} F_{d17}$

(19) $N_{d4} \rightarrow N_{d13} A_{d5} N_{d14}$

(20) $N_{d8} \rightarrow N_{d15} A_{d5} N_{d14}$

(21) $N_{d13} \rightarrow F_{d18} A_{d10} F_{d19}$

(22) $N_{d15} \rightarrow F_{d20} A_{d11} F_{d21}$

(23) $N_{d14} \rightarrow F_{d22} A_{d6} N_{d15}$

(24) $N_{d15} \rightarrow F_{d23} A_{d9} F_{d24} A_{d8} F_{d25}$

(25) $N_{d15} \rightarrow F_{d26} A_{d7} F_{d27}$

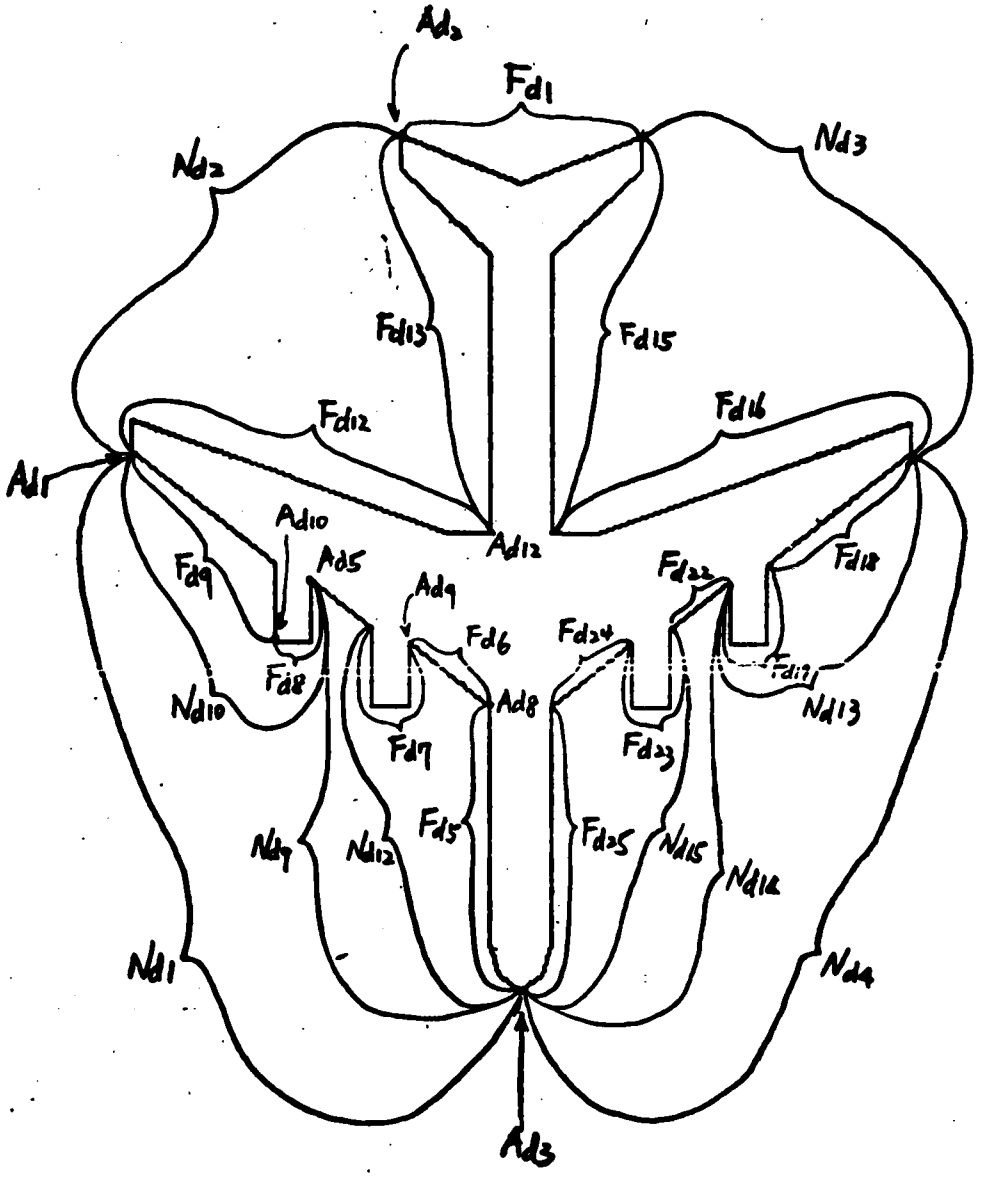

Ad2
Fd1
Nd2
Nd3
Fd13
Fd15
Fd12
Fd16
Ad1
Ad10
Ad12
Fd9
Ad5
Ad9
Fd22
Fd18
Fd6
Fd24
Fd8
Fd21
Nd10
Ad8
Nd13
Fd7
Fd23
Nd9
Nd12
Fd5
Fd25
Nd15
Nd16
Nd1
Nd4
Ad3

Fig. 4.3

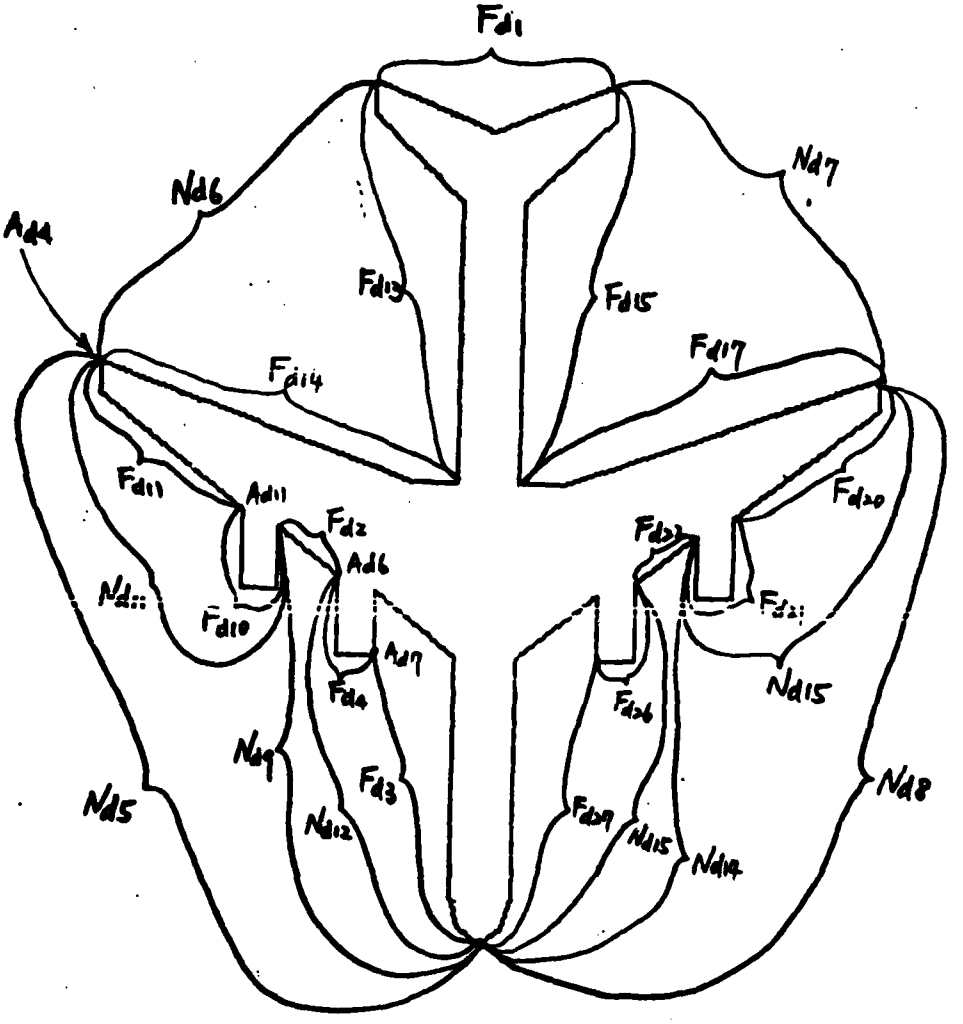

Fd1
Nd6
Nd7
Ad4
Fd13
Fd15
Fd17
Fd14
Fd11
Ad11
Fd2
Ad6
Fd20
Fd21
Nd11
Fd10
Ad7
Fd4
Nd15
Fd16
Nd9
Fd3
Nd5
Nd8
Nd12
Fd19
Nd18

Fig. 4.4

The shape grammar for BAC 1-11, $G_c = (V_c, T_c, P_c, S_c)$

$V_c = \{ S_c, N_{ci} \mid 1 \leq i \leq 8 \}$

$T_c = \{ F_{cj}, A_{ck} \mid 1 \leq j \leq 15, 1 \leq k \leq 7 \}$

P_c :

(1) $S_c \longrightarrow N_{c1}A_{c1}N_{c2}A_{c2}F_{c1}A_{c2}N_{c3}A_{c1}N_{c4}A_{c3}$

(2) $S_c \longrightarrow N_{c2}A_{c2}F_{c1}A_{c2}N_{c3}A_{c1}N_{c4}A_{c3}N_{c1}A_{c1}$

(3) $S_c \longrightarrow F_{c1}A_{c2}N_{c3}A_{c1}N_{c4}A_{c3}N_{c1}A_{c1}N_{c2}A_{c2}$

(4) $S_c \longrightarrow N_{c3}A_{c1}N_{c4}A_{c3}N_{c1}A_{c1}N_{c2}A_{c2}F_{c1}A_{c2}$

(5) $S_c \longrightarrow N_{c4}A_{c3}N_{c1}A_{c1}N_{c2}A_{c2}F_{c1}A_{c2}N_{c3}A_{c1}$

(6) $S_c \longrightarrow N_{c6}A_{c2}F_{c1}A_{c2}N_{c7}A_{c4}N_{c8}A_{c3}N_{c5}A_{c4}$

(7) $S_c \longrightarrow N_{c8}A_{c3}N_{c5}A_{c4}N_{c6}A_{c2}F_{c1}A_{c2}N_{c7}A_{c4}$

(8) $N_{c1} \longrightarrow F_{c2}A_{c5}F_{c3}$

(9) $N_{c5} \longrightarrow F_{c2}A_{c5}F_{c4}$

(10) $N_{c2} \longrightarrow F_{c5}A_{c6}F_{c6}A_{c7}F_{c7}$

(11) $N_{c6} \longrightarrow F_{c8}A_{c6}F_{c6}A_{c7}F_{c7}$

(12) $N_{c3} \longrightarrow F_{c9}A_{c7}F_{c10}A_{c6}F_{c11}$

(13) $N_{c7} \longrightarrow F_{c9}A_{c7}F_{c10}A_{c6}F_{c12}$

(14) $N_{c4} \longrightarrow F_{c13}A_{c5}F_{c14}$

(15) $N_{c8} \longrightarrow F_{c15}A_{c5}F_{c14}$

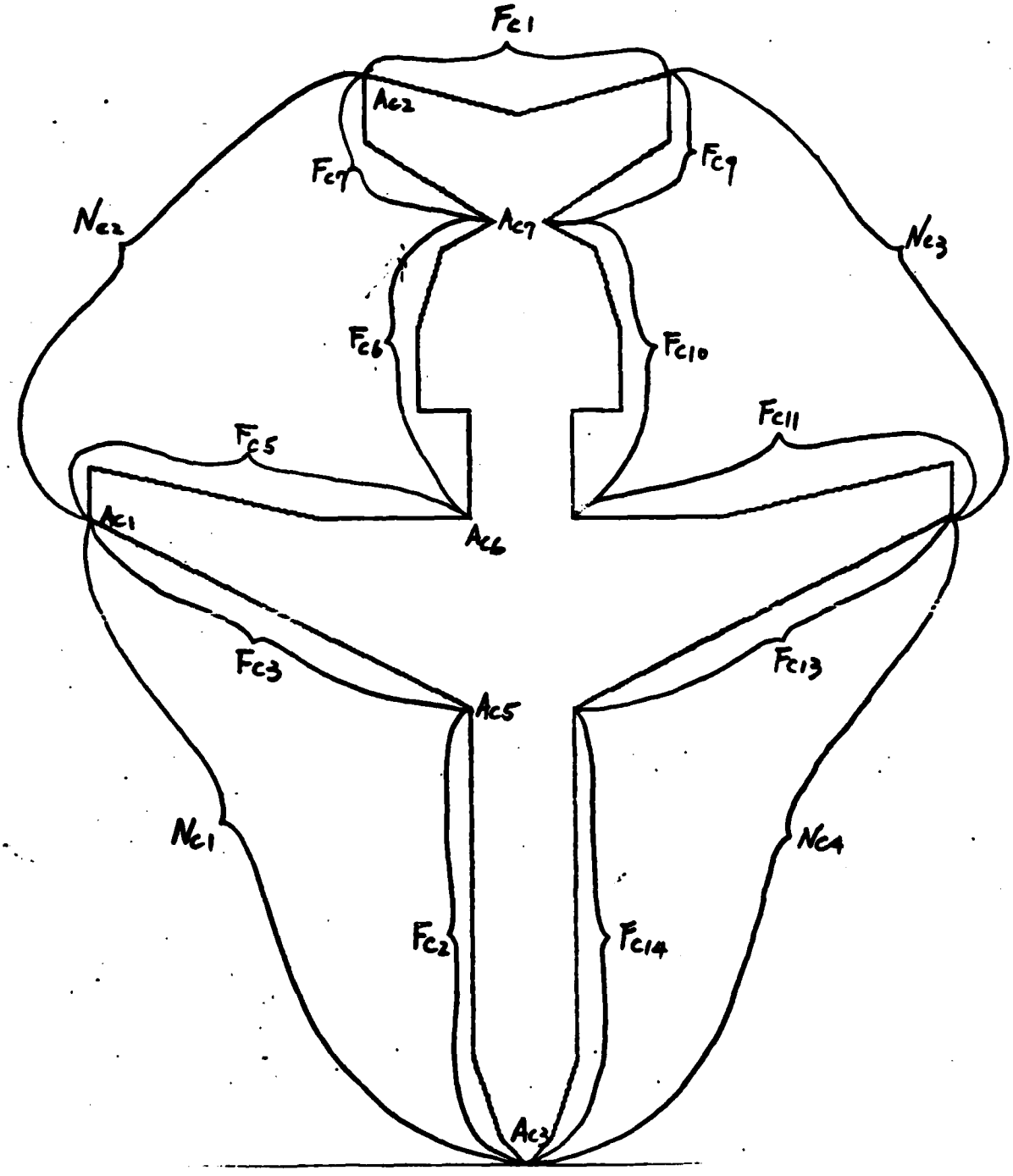

Fc1
Ac2
Fc7
Fc9
Nc2
Ac7
Nc3
Fc6
Fc10
Fc11
Fc5
Ac1
Ac6
Fc3
Fc13
Ac5
Nc1
Nc4
Fc2
Fc14
Ac3

Fig. 4.5

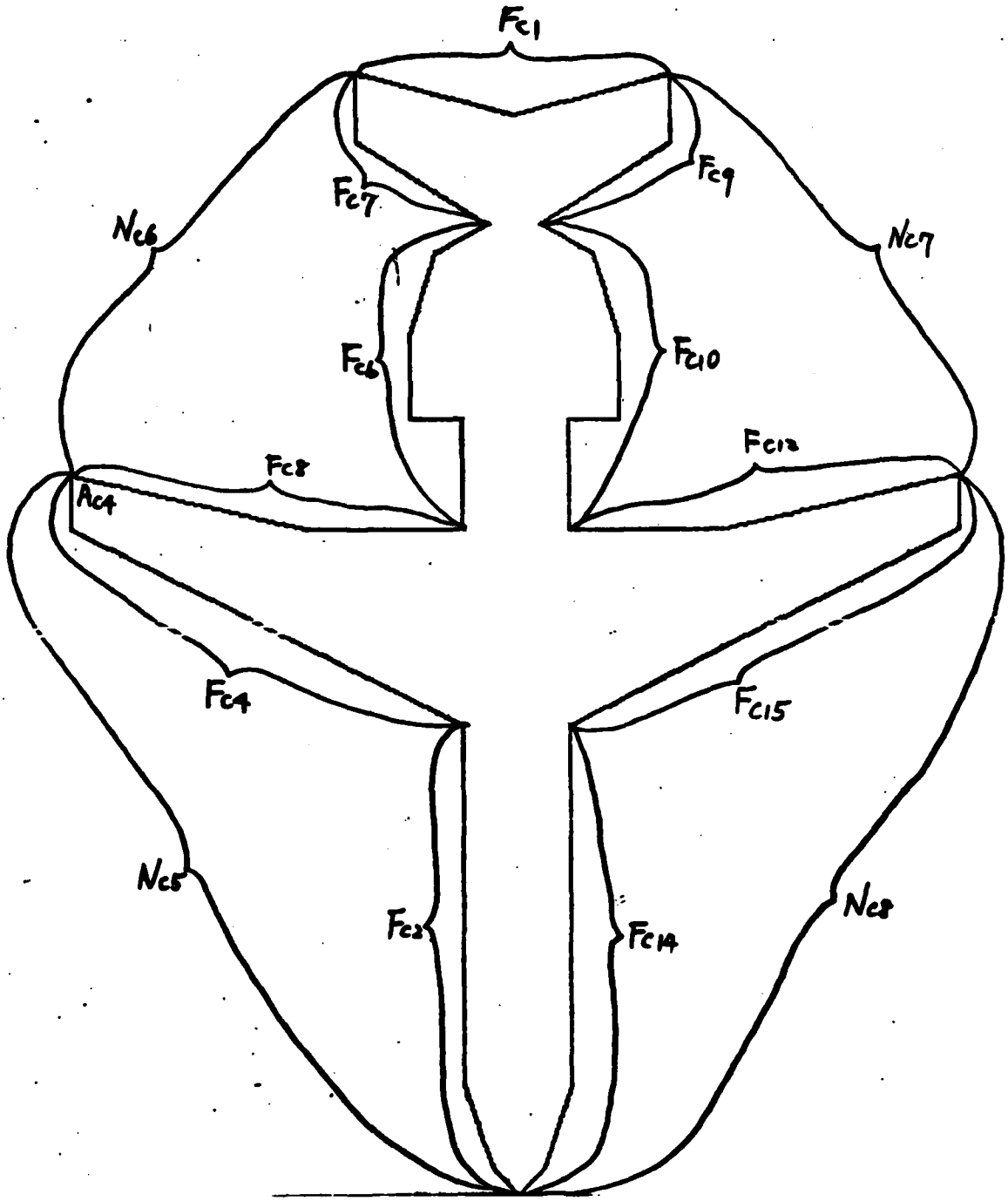

Fc1
Fc7
Fc9
Nc6
Nc7
Fc6
Fc10
Fc8
Fc12
Ac4
Fc4
Fc15
Nc5
Nc8
Fc2
Fc14

Fig. 4.6

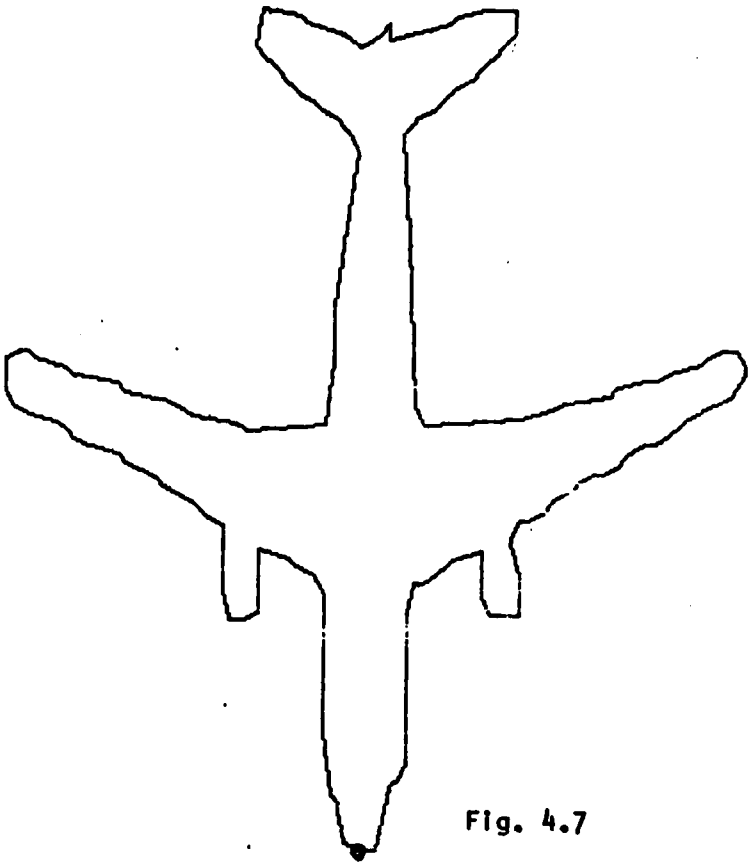


Fig. 4.7

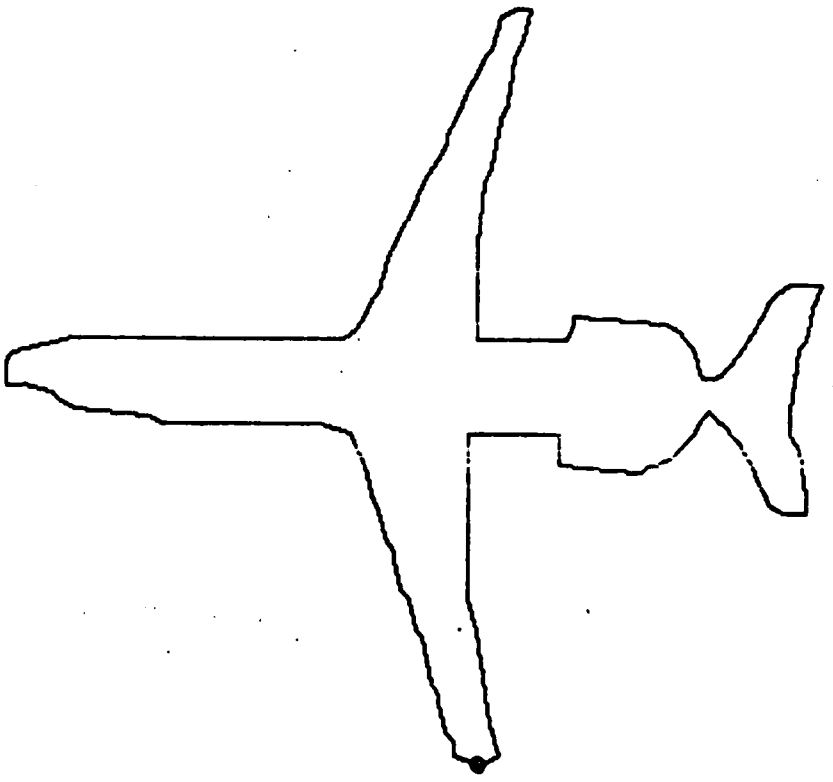


Fig. 4.8

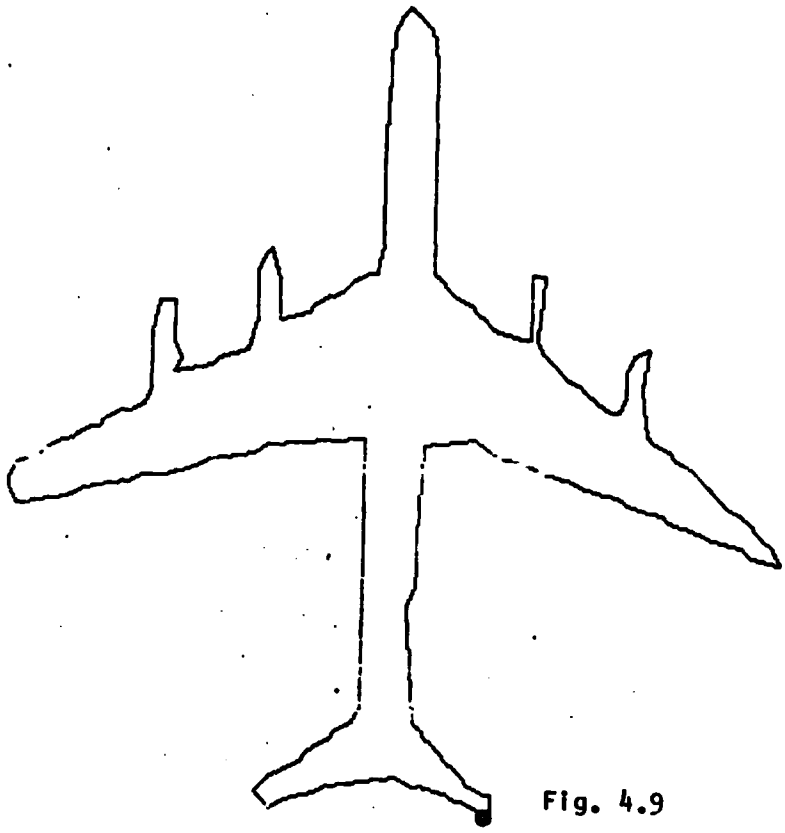


Fig. 4.9

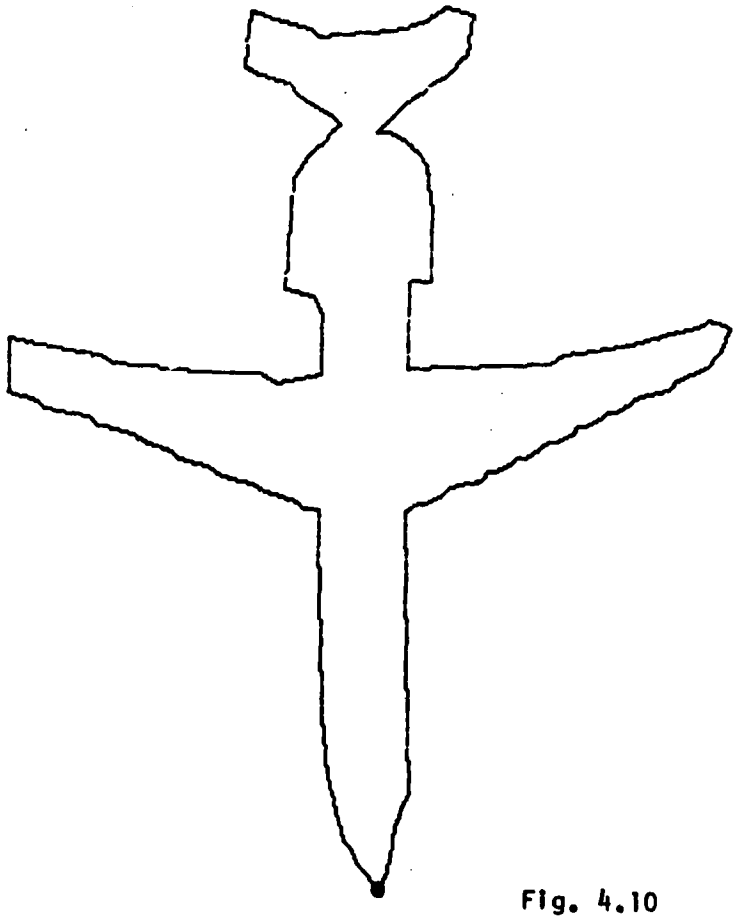


Fig. 4.10

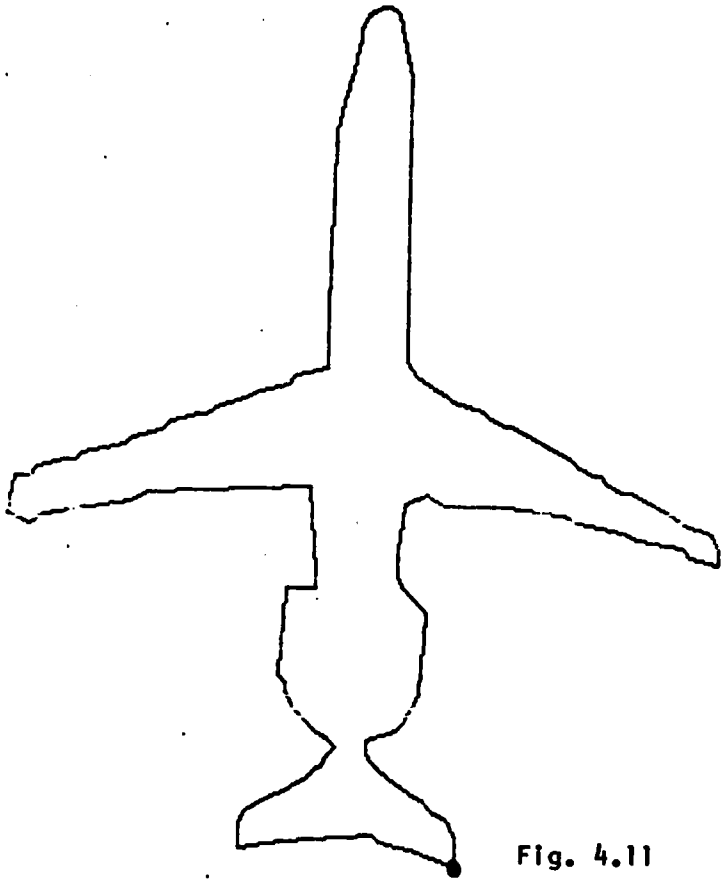


Fig. 4.11

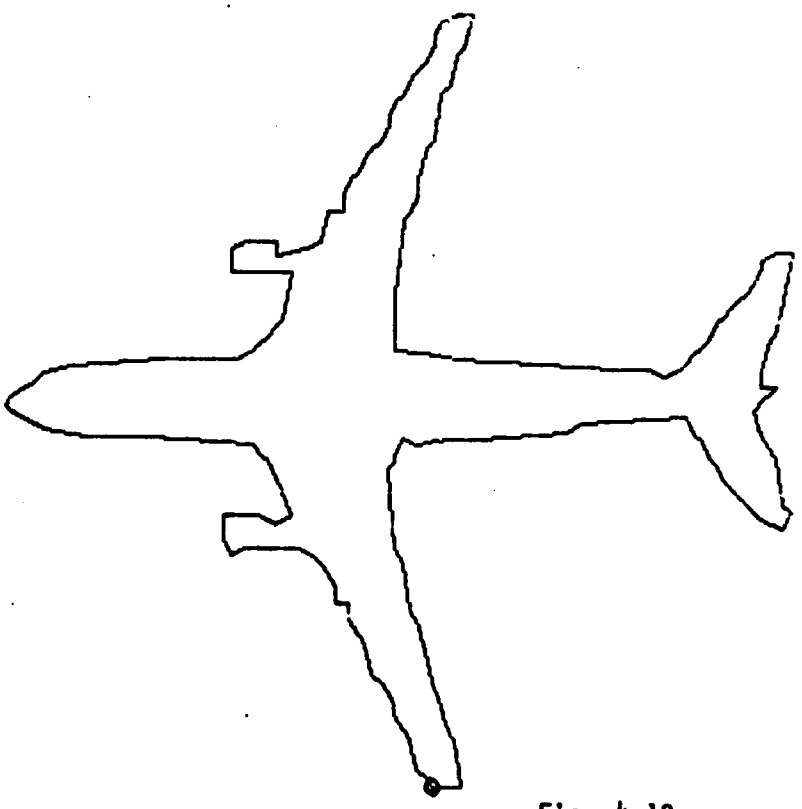


Fig. 4.12

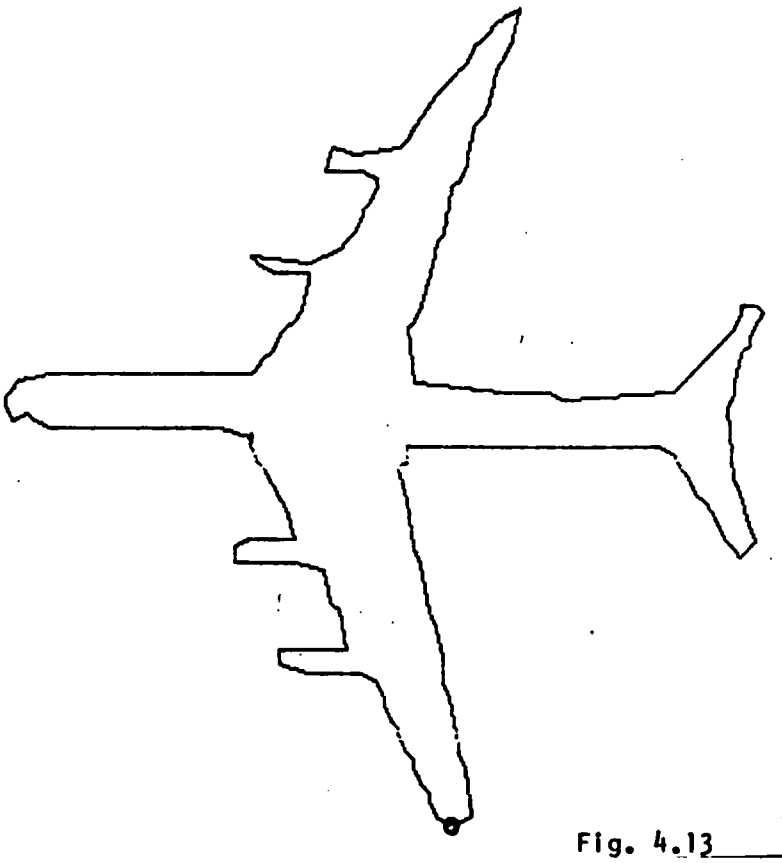


Fig. 4.13

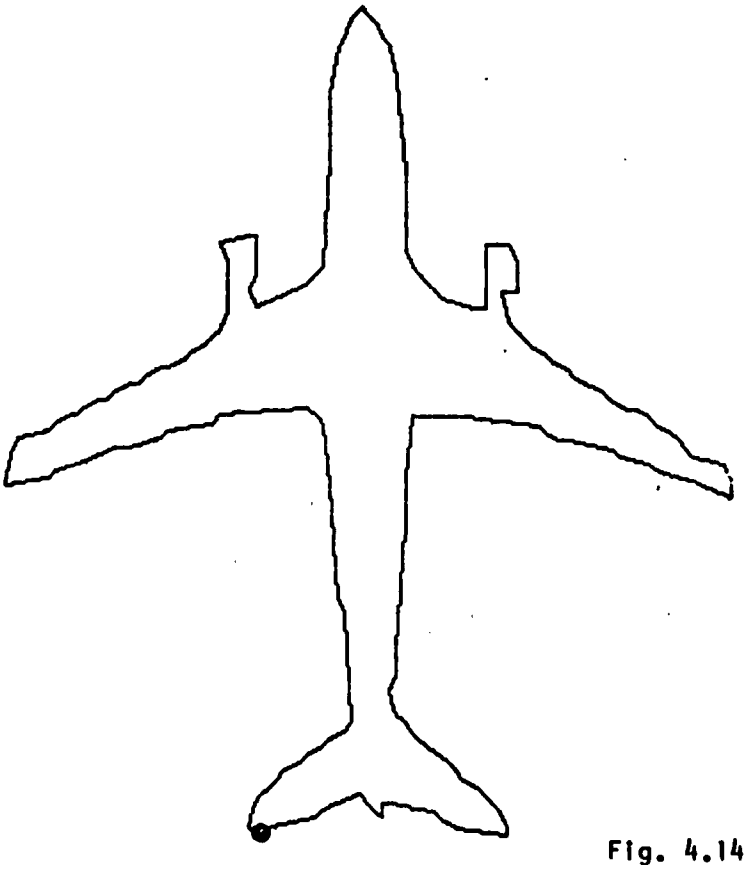


Fig. 4.14

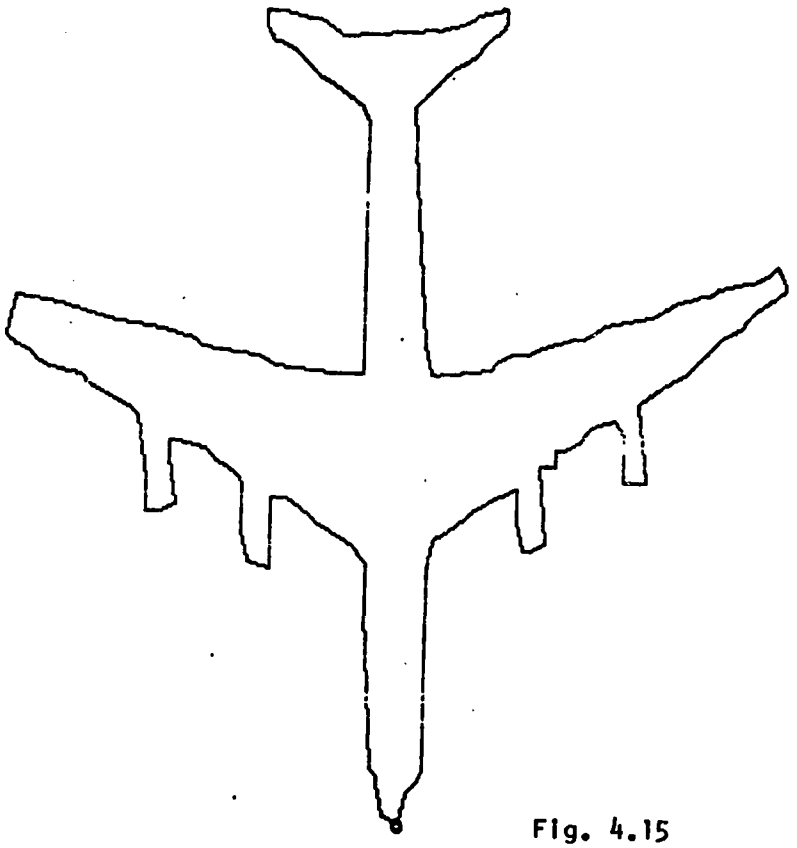


Fig. 4.15

IMAGE CLASSIFICATION EMPLOYING STATISTICAL CONTEXT

E. F. Kit and P. H. Swain

INTRODUCTION

Useful information from remote sensing data is obtained by accurately classifying each region in the image. In the past, work dealing with the computer analysis of image data collected from high-flying aircraft and earth orbiting satellites has focused on classifying each point in the image (on the ground) in terms of its spectral variation in electromagnetic field strength. Spectral classifiers generally classify a single image point based solely on data associated with that point.

Recently the focus has been broadened to employ spatial relationships in the data. Even an amateur photo interpreter would concur that this information component is substantial. For example, automobiles occur more often on highways than in water, and boats are more often in water than on highways. Then, even if their spectral appearance is the same, boats can be discriminated from automobiles if their context is considered. The data immediately surrounding an image point is intimately associated with it and can be used to throw light upon its true nature.

One approach to the statistical treatment of context was suggested by Welch and Salter [1]. However their basic assumption (made in an effort for a practical solution) was that contextual relationships between nonadjacent cells are negligible. They later went on to make extensive comparisons between the recognition performance of a four neighbor rule and an eight neighbor rule.

The model we have formulated is based on a significantly different approach to spatial dependencies. Contextual configurations of arbitrary shape (see Fig. 1) can be dealt with by our model and statistical dependencies

extracted from a typical scene used to classify a new scene are less complex and more readily determined than those required by the Welch and Salter approach. The block diagram shown in Fig. 2 summarizes our initial experiment and in the next section we focus on context extraction.

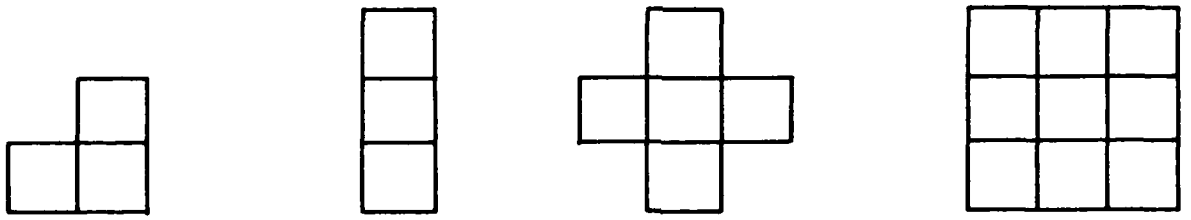


Fig. 1. Contextual Neighborhoods of Arbitrary Shape

Context Extraction

Recall the context classification class discriminant function as developed in the previous progress report:

$$\sum_{\substack{\underline{\theta}, \\ \theta_1=\theta}} \left[\prod_{i=1}^{p} f_{\theta_i}(X_i) \right] G^p(\underline{\theta})$$

where θ_i and X_i are respectively the state and measurement of the ith cell in the p array and θ_1 and X_1 refer to the cell being classified.

Tabulation of the frequency distribution, $G^p(\theta)$, for typical scenes of interest began as a challenging problem because of the growth of memory requirements due to large window size (context neighborhood) and large number of classes. Consider the pixel array in Fig. 3 and the simple case where the context window size is three and number of classes is two as shown in Fig. 4.

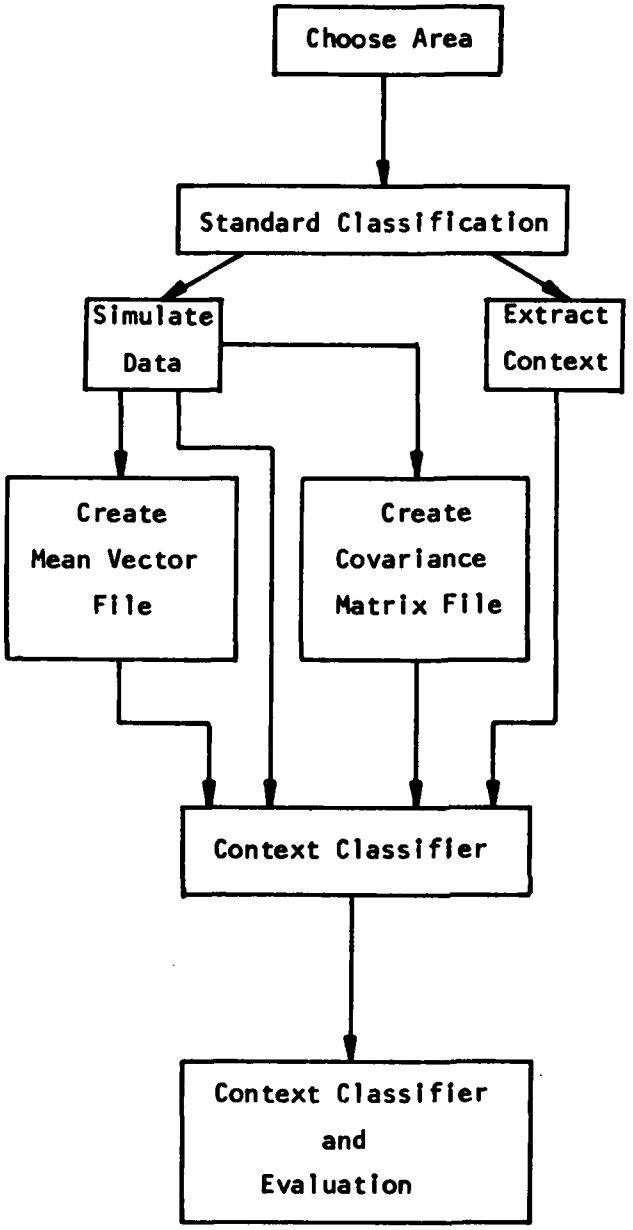


Fig. 2. Block Diagram of Initial Experiment

To tabulate the frequency distribution, the specified window is placed over the pixel array beginning in the upper left corner. A count associated with this combination of classes inside the window is incremented. The window is moved one pixel to the right, and the process repeated until the specimen pixel array has been exhausted.

The simplest solution is to reserve memory for each possible combination and to store the associated frequency. Consider the fairly typical situation, however, in which the window size is nine (three by three neighborhood) and the number of classes is seventeen. Each three by three block can take one of 17 to the ninth power (or 120000000000) possible configurations and the simple approach is obviously infeasible.

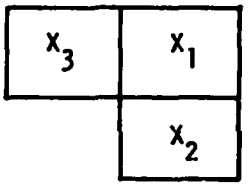
A	F	A	A
F	A	F	F
F	A	A	F
F	A	A	A

A and F represent distinct classes

Fig. 3. Sample Pixel Array for Frequency Distribution Determination

Note that a given scene, having finite size, must contain only a small fraction of this huge number. For example, consider classifying a 200 pixel by 200 pixel scene. In the worst case there are 40000 different combinations present. Many of these 40000 will be repeated often, further reducing the number of unique combinations. Only combinations which actually occur need be stored.

Window Size = 3
Number of Classes = 2



Possible Configurations

X_1	X_2	X_3
A	A	A
A	A	F
A	F	A
A	F	F
F	A	A
F	A	F
F	F	A
F	F	F

Fig. 4. Sample Context Problem

Figure 5 summarizes the presently implemented approach. Each time a combination is obtained, the binary tree is traversed until the combination is found or until a null node is encountered. If the combination is already contained in the tree, the count associated with that combination is incremented. Otherwise this is the first time this combination has occurred, and therefore a new node is added to the tree and its associated count is initialized to one.

For our first frequency tabulation, the frequency distribution was determined from a classification 114 columns by 251 lines. The total number of nodes in the final binary tree was found to be 18502. The ratio of the actual number of unique combinations to the possible number of unique combinations (a function of scene size) provides an indication of the "amount" of context in the scene. In this case our "scene context ratio" was:

$$\text{scene context ratio} = \frac{18502.}{114 \times 251} = .65$$

1. Input
   a. Classification Results File
   b. Window Configuration
   c. Image Size
2. Get Combination
3. Tree
   Combination Found
     a. Increment Count
   Null Node
     a. Create Node
     b. Count = 1
4. If More Combinations Go to 2
5. Print Tree Using Inorder Traversal

Fig. 5. Binary Tree Implementation

The scene context ratio is always greater than zero and less than or equal to one. For a scene with frequently occurring combinations, the numerator will tend to be small, resulting in a low scene context ratio and indicating a high-context scene.

Context Classification and Evaluation

The Context Classification algorithm is a supervised classification algorithm since it requires training samples representative of each class of interest. In our approach, the classes are assumed characterizable by multivariate normal probability density functions. Therefore the training samples are used to estimate mean vectors and covariance matrices for each class.

The goal of the Context Classifier is exactly the same as the familiar Maximum Likelihood Classifier (MLC) and is shown in Fig. 6.

$$\mathbb{R}^{nxmxv} \rightarrow \mathbb{R}^{nxm}$$

Fig. 6. MLC and Context Classification Goal

The number of lines, columns, and channels in the image are represented by n, m, and v respectively. These classifiers are responsible for assigning a given input pattern to a class by comparing the feature vector with a set of training patterns. The common performance measure among Context and conventional ML Classifiers is "minimum probability of error".

The important input difference which distinguishes the Context Classifier is shown in Fig. 7.

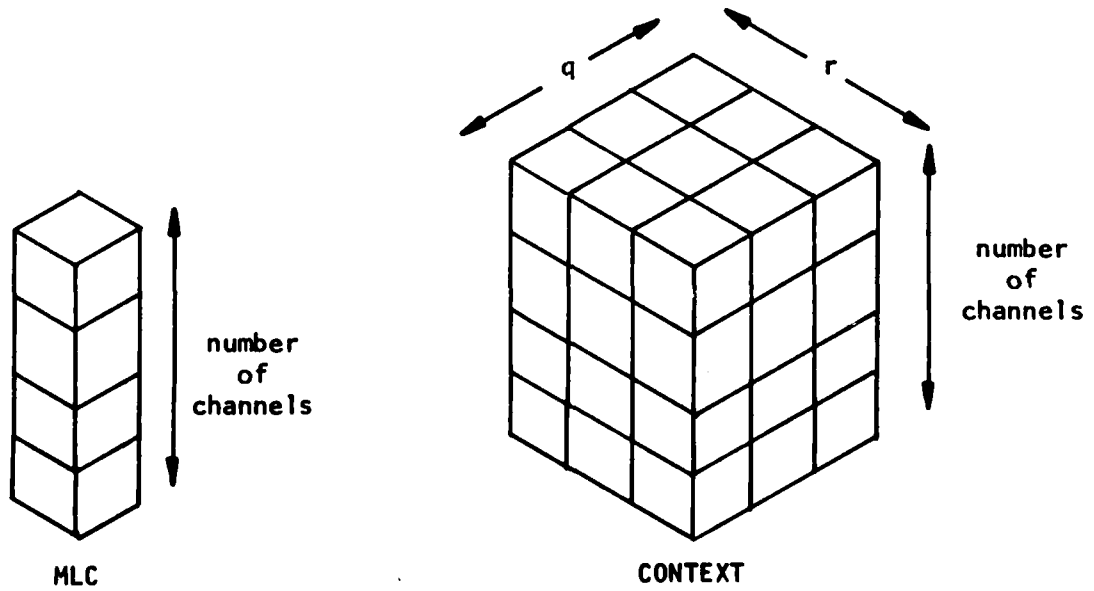


Fig. 7 Input Difference Between Context and ML Classifiers

For the full window shown, where $q = r = 3$, the Context Classifier inputs nine times the information per point of MLC. As mentioned earlier, any subset of the full window may be easily used, and if $q = r = 1$, the Context Classifier is precisely the same as the MLC.

Software supporting the Context Classifier has been developed and is currently under test using multispectral remote sensing imagery. Results of applying this approach to classification, including qualitative and quantitative comparisons with MLC results, will appear in the next progress report.

REFERENCE

[1] J. R. Welch and K. G. Salter, "A context algorithm for pattern recognition and image interpretation," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-1, No. 1, January 1971.

A SPATIAL STOCHASTIC MODEL FOR CONTEXTUAL PATTERN RECOGNITION

T. S. Yu and K. S. Fu

Abstract

A contextual classification algorithm using a spatial stochastic model (Markov random field) is proposed. The requirements for the joint probability function on the two-dimensional lattice are discussed. The distinction between the spatial correlation context and the transition probability context is made. Conceptually clear procedures for construction of the model are given. The coding technique for parameter estimation is presented. An extension of the model in the multivariate site variable case is derived to handle the multispectral satellite data. Experiments with remote sensing data are performed and results are compared with simple (no context) rule results. Less frequently occurring classes such as highways and commercial areas were found to be classified better using the contextual algorithm with only a reasonable increase in time computation.

1. Introduction

Compound decision theory has enabled us to introduce contextual information into the statistical decision procedure for image interpretation. Based on initial success in using context information [1], [2], the investigation has been extended and is going beyond the simple (no context) classification to improve the decision procedure in areas where the need is clear and the possibilities are promising. The outcome of this investigation has been a conclusive demonstration of the feasibility of applying a spatial stochastic model to analyze digital satellite data. The effectiveness of this model was demonstrated by the minor increase in computation

time and the improved classification results. Some modifications and extensions of the model were specifically developed or substantially refined during this investigation to cover more general situations. A wide variety of agricultural areas and urban residential areas were chosen for experiments. Other classes in the satellite data include pastures, forests, water bodies and highways.

2. Proposed Approach

Under appropriate assumptions, the derived compound decision rule [2] is to choose action $a \in A$ to minimize

$$\sum_{\theta_k} L(\theta_k, a) P(x_k/\theta_k) G(\theta_k) \prod_{i=1}^{4} \sum_{\theta_{k_i}} P(x_{k_i}/\theta_{k_i}) P(\theta_{k_i}/\theta_k) \qquad (1)$$

where $\theta_k \in \theta$ is the kth class, x_k is the pattern vector of cell k, G is the a-priori probability function and L is the loss function. Suppose we let $G_1(\theta_k) = G(\theta_k) \prod_{i=1}^{4} \sum_{\theta_{k_i}} P(x_{k_i}/\theta_{k_i}) P(\theta_{k_i}/\theta_k)$. The decision rule (1) becomes

$$\sum_{\theta_k} L(\theta_k, a) P(x_k/\theta_k) G_1(\theta_k) \; . \qquad (2)$$

Rule (2) is of the same form as the simple decision rule except for the a-priori probability $G_1(\theta_k)$. Note that $G_1(\theta_k)$ is now the probability of occurrence of the four nearest neighboring states of θ_k together with that of θ_k. Since the neighboring states are unknown, we have to obtain the probability of the neighboring patterns belonging to each individual class and of each individual class being a neighbor of θ_k. The four multipliers in the product term in (1) represent the contextual contributions from the four neighboring cells. Context information in terms of directional transition

probabilities is introduced in this decision scheme.

The first problem with this scheme is the validity of the estimated directional transition probabilities from training data. Usually the ground truth information is never known with certainty, especially with satellite data.

Another problem is that we assume the contributions from the four neighbors are independent. In real situations this is not generally true, for example, agricultural areas are usually present on both sides of a river.

Instead of using the transition probabilities, we are interested in using the correlations between different patterns since the correlation-type context does not require us to know the true classes of the patterns. In particular, we would like to obtain the joint density function of θ_k and its four nearest neighbors (Figure 1) because the joint distribution function

$$
\begin{array}{ccccc}
 & & K_{11} & & \\
 & K_7 & K_2 & K_6 & \\
K_{12} & K_3 & K & K_1 & K_{10} \\
 & K_8 & K_4 & K_5 & \\
 & & K_9 & &
\end{array}
$$

Figure 1. Order of neighborhood of cell K.

contains all the correlations between all pairs of cells. In later classification processes we shall concentrate on the observation vector $X_k = (x_k, x_{k_1}, x_{k_2}, x_{k_3}, x_{k_4})$ instead of just on x_k to classify cell K.

Notice that this joint density function must be defined consistently. When we try to classify cell K_1, we need to use the same joint density structure as when we classify cell K. This implies that the marginal joint

distribution functions of cells

$$\begin{array}{ccccc} & & K_2 & k_2 & \\ K_3 & K_{k_3} & K_{1_k} & & k_1 \\ & & K_4 & k_4 & \end{array}$$

Figure 2. Two NB (nearest neighbor) system samples overlapping.

x_k, x_{k_1} should be the same as those of cells x_k, x_{k_3} (see Figure 2). In the Gaussian case, this means that the variances of cells K, K_1, K_3 are all the same and that the correlations between K, K_1 are the same as those between K, K_3. In general, we must require that the labelling of the different cells should not affect the definition of the joint density functions for these cells. Also we must require that all the variance-covariances be distance dependent but not position dependent. For example, (see Fig. 1) the covariances between cells K, K_8 should be the same as those between cells K_1, K_4 since the distance between either pair is the same. This requirement is commonly known as "stationarity".

We need to construct a two-dimensional random process with the stationarity property. In particular, we are mostly interested in the Gaussian plane process because the gaussian density functions can be specified by the second moments. This means that we should be able to specify covariances between any pair of cells (or sites) in the lattice after constructing the stationary random process.

3. A Markov Process on a Torus

The procedure for constructing a two-dimensional stationary process is similar to the one-dimensional case. Here we first construct a similar process on a lattice torus in which all random variables, x_{mn}, are identically normally distributed and satisfy a Markovian property. Then we let the size of the torus tend to infinity to form the stationary plane process. Let the torus be defined by all pairs of integers (j,k), $j=0,\ldots,p-1$, $k=0,\ldots,q-1$ where p and q are identified with zero. There are pq random variables x_{jk} at points (j,k) and we suppose that they have a joint normal distribution with the density function

$$p(\cdot) = K \exp\left[\frac{-1}{2}\left(\Sigma\, x_{jk}^2 + a\,\Sigma\, x_{jk}(x_{j+1,k}+x_{j-1,k}+x_{j,k+1}+x_{j,k-1})\right)\right] \tag{3}$$

where $|a| < \frac{1}{4}$, and K is a constant. The quadratic form inside the brackets can be shown to be positive definite.

This torus process has been discussed separately by Besag [3] and Moran [4]. Besag presented a conditional probability approach to specify the spatial process and derived the joint density function which is equivalent to (3). Here we follow Moran's procedure for constructing the desired plane process since it is brief and clear.

We now find the variance-covariance matrix of the distribution of the X's by inverting the matrix of the coefficients of the quadratic form in (3). We write the pq random variables as a vector X equal to $(x_{00},x_{01},\ldots,x_{0,q-1}, x_{11},\ldots,x_{1,q-1},\ldots,x_{p-1,q-1})$. We write the matrix Q as $(C_{u_1-u_2,v_1-v_2})$ where $C_{u_1-u_2,v_1-v_2}$ is the element in row u_1q+v_1 and column u_2q+v_2 where $u_j = 0,1,\ldots,p-1$ and $v_j = 0,1,\ldots,q-1$ and their sums and

differences are taken modulo (p) and (q) respectively.

We construct the matrix Q as follows. Consider the p x p circulant matrix

$$\begin{bmatrix} x & y & 0 & \cdot & \cdot & \cdot & y \\ y & x & y & 0 & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ y & 0 & 0 & \cdot & \cdot & y & x \end{bmatrix}$$

and replace the x, y and zero by A, B and zero q x q matrices respectively where A is the p x p matrix

$$\begin{bmatrix} 1 & a & 0 & \cdot & \cdot & a \\ a & 1 & a & \cdot & \cdot & 0 \\ 0 & a & 1 & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a & 0 & \cdot & 0 & a & 1 \end{bmatrix}$$

and B is a p x p diagonal matrix with a's down the main diagonal. Notice that the Q matrix here is exactly the same as Besag's B matrix in his paper.

Next we can do a unitary transformation to diagonalize Q in order to find its inverse. We only give the final result here. Write $Q^{-1} = (b_{st})$ where $s = u_1 q + v_1$, $t = u_2 p + v_2$. Then

$$b_{st} = \sum_m (pq)^{-1} \exp 2\pi i\{u_1 u_3 p^{-1} + v_1 v_3 q^{-1}\}$$

$$\cdot \{1 + 2a \cos 2\pi v_3 q^{-1} + 2a \cos 2\pi u_3 q^{-1}\}^{-1}$$

$$\cdot \exp -2\pi i\{u_2 u_3 p^{-1} + v_2 v_3 q^{-1}\} \qquad (4)$$

where $m = u_3 q + v_3$.

This is real and is the variance-covariance matrix of the random variables on the torus.

4. A Markov Process on a Plane Lattice

Our goal is to construct a stationary Markovian Gaussian process on the lattice formed by all positive and negative integers (m,n). To do this it is sufficient to define such a process for all finite sets of such points provided the resulting distribution is invariant under translation (consistency requirement). It is therefore sufficient to consider any fixed finite set of points on the torus lattice and then let p and q tend to infinity. Writing $\exp 2\pi i\, u_3 p^{-1} = z_1$ and $\exp 2\pi i\, u_3 q^{-1} = z_2$ and letting p,q tend to infinity, we find that the covariance $V_{s,t}$ between $x_{m,n}$ and $x_{m+s,n+t}$ is

$$\frac{-1}{4\pi^2} \int_{|z_1|=1} \int_{|z_2|=1} \frac{z_1^{s-1}\, z_2^{t-1}}{[1+a(z_1+z_1^{-1})+a(z_2+z_2^{-1})]} dz_1\, dz_2 \tag{5}$$

with the convergence being uniform on the finite set of points.

We can obtain the covariances explicitly from the integral

$$V_{s,t} = \frac{1}{4\pi^2} \int_0^{2\pi} \int_0^{2\pi} \frac{\cos s\theta_1 \cos t\theta_2}{[1+2a\cos\theta_1+2a\cos\theta_2]}\, d\theta_1 d\theta_2 \tag{6}$$

V_{00} can be obtained [4] analytically as

$$v_{0,0} = 2\pi \int_0^{\frac{\pi}{2}} (1-16\, a^2 \sin^2\theta)^{\frac{-1}{2}}\, d\theta \tag{7}$$

which is $(2\pi)^{-1}$ times the complete elliptic integral of the first kind, tabulated in Abramowitz and Stegum [5].

$$V_{0-1} = V_{01} = V_{10} = V_{-10} = \frac{1}{4} a^{-1}(1-V_{00})$$

Other values of $V_{s,t}$, which are known as the lattice Green's function, are much more complicated to evaluate. We used summations to approximate the integrals in (6) to obtain V_{02} and V_{11} for our NB system.

5. Estimation of the Spatial Correlation Parameter

The parameter "a" in (3) is unknown and must be estimated from training samples. There is more than one way to do the parameter estimation, for example, see [3] [6]. However we will describe a coding method which is very flexible and simple.

We have mentioned that Besag used a conditional probability approach to define the torus process. Specifically he derived the most general (but consistent) conditional distribution function and went on to arrive at the joint distribution for all cells in the rectangular lattice. The conditional distribution he derived was $P(x_k/\text{all other sites}) = (2\pi\sigma^2)^{\frac{-1}{2}} \exp[\frac{-1}{2}\sigma^{-2}\{x_k-\mu_k \Sigma \beta_{kj}(x_j-u_j)^2\}]$. The Markov definition used for defining this process was

$$P(x_k/\text{all other cells})$$

$$= P(x_k/4 \text{ nearest neighboring cells k})$$

In order to fit a first order scheme, we begin by labelling the interior sites of the lattice, alternately x and •, as shown in Figure 3. It is

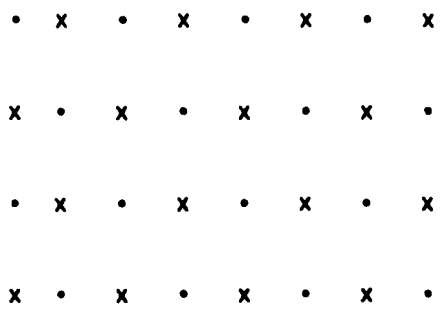


Figure 3. Coding pattern for a first-order (NB) scheme.

immediately clear that, according to the first-order Markov assumption the variables associated with the x sites, given the observed values at all other sites, are mutually independent. This results in the simple conditional likelihood,

$$\Pi\, p(x_{i,j}/x_{i-1,j}, x_{i+1,j}, x_{i,j-1}, x_{i,j+1}) \tag{9}$$

for the x site values, the product being taken over all x sites. The conditional Maximum Likelihood Estimate of the unknown parameter can be obtained in the usual way.

The defined process requires that every variate (associated with each cell) be identically normally distributed. However, in practical situations such as our satellite data, there are many classes appearing in the image and they are statistically characterized by different mean vectors and covariance matrices. For the single-variate site variable case discussed so far we may easily normalize each pattern so that every variate has zero mean and unit variance. In the multivariate-site variable case, we have to take into account other considerations. These are discussed below.

6. Multivariate Site Variable Extension

We will consider the two-variate site variable case and then the more than two-variate case can be generalized by straightforward extension. Imagine that we have, at each site, two variates for the NB scheme (Figure 4a).

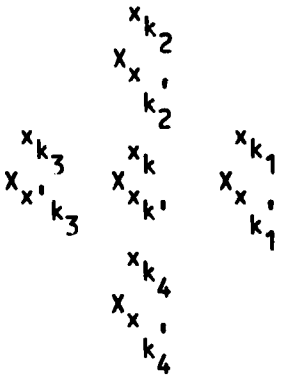


Figure 4a

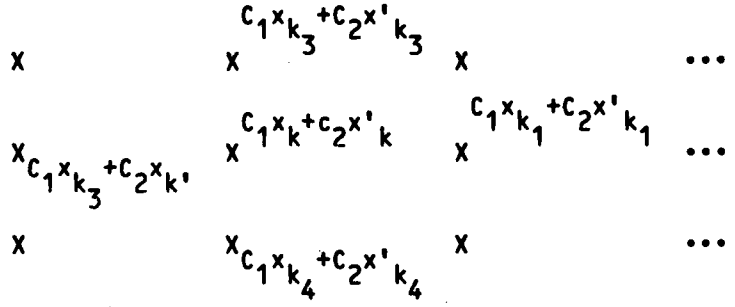


Figure 4b.

It is now necessary to construct a 10 x 10 covariance matrix to incorporate all kinds of correlations between these 10 variates. However, only the correlation between one variate of one site and another variate of another site is unknown. We have to find this correlation to form our covariance matrix.

Consider the 1 dimensional NB scheme in Figure 4b where every variate is obtained from a linear combination of the two variates at every site in Figure 4a. The process in Figure 4b must preserve the original Markov property of the one-dimsional process discussed earlier. In other words, all

the variances and covariances between sites in Figure 4b can only undergo a scale change from the one-dimensional case such that the overall covariance structure has the same form and can be obtained from (6) with some other correlation parameter "a".

The variance of site k in Figure 4b is $c_1^2+c_2^2+2c_1c_2E[x_k,x_k']$. Therefore the variance is increased by a factor of $c_1^2+c_2^2+2c_1c_2E[x_k,x_k']$ (remember the original variate had unit variance). The covariance between site k and any other site say ℓ, is $E[x_k,x_\ell]$ $\{c_1^2+c_2^2+2c_1c_2 \frac{E[x_k,x_k']}{E[x_k,x_\ell]}\}$ since $E[x_k,x_\ell] = E[x_k',x_\ell']$ and $E[x_k,x_\ell'] = E[x_\ell,x_k']$.

In order to have the same scale change for the covariance as for the variance, we must impose

$$E[x_k,x_k'] = \frac{E[x_k,x_\ell']}{E[x_k,x_\ell]} \tag{10}$$

This will enable us to construct our 10 x 10 covariance matrix. Another consideration worth noting is the normalization process for the multivariate case. An orthonormal transformation is required to transform the variates into zero mean and an identity matrix as their covariance matrix. This is also called the "Whitening process" [7]. The transformation matrix is $\Lambda^{-\frac{1}{2}}\Phi^T$ where Λ is the eigenvalue matrix and $\Phi = [\phi_1,\phi_2]$ is the eigenvector matrix for the covariance matrix of the particular class the pattern belongs to. After the orthonormal transformation, each site will have two uncorrelated variates with zero means and unit variance. This will make our overall (10 x 10) covariance simpler because of (10). The 10 x 10 covariance matrix for the 10 variates in the NB system (Fig. 4a) would be

$$\Sigma_{10x10} = \begin{bmatrix} \Sigma_B & 0 \\ 0 & \Sigma_B \end{bmatrix} \tag{10}$$

where $\sum_B$ is the 5 x 5 covariance matrix of the 1-dim NB system with elements evaluated from (6) once "a" has been estimated.

It is not hard to show that $\sum_{10x10}$ is positive definite since we have guaranteed that $\sum_B$ is positive definite. Generalization of the more than two variate site variable case can easily be carried out.

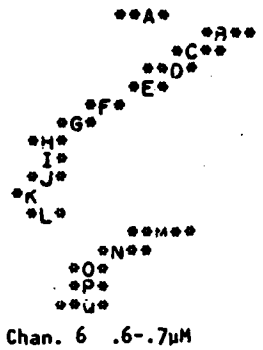
7. Experimental Results

The classification procedure (contextual) requires us to know the four neighboring classes in order to evaluate the joint density function. We may not need the true class but some class knowledge of those neighbors ought to be known. We obtained these classes by doing a "preclassification" using the simple (no context) decision rule. The contextual classification will then be called the "postclassification".

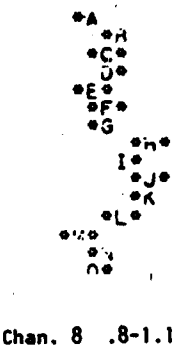
The data used for the experiment is the satellite data of Lafayette, IND under Run No. 72053609 provided by LARS*. Usually four spectral bands are used to collect Satellite (LANDSAT) data (except for Temporal data). However, it was found that two bands (one from the infrared region and one from the visible region) are enough to produce maximum feature separation. The two bands used are band 3 (0.7-0.8 μM), and band 6(0.6-0.7 μM).

The lack of large homogeneous areas in the LANDSAT data provides great difficulty for training and testing purposes. There are seventeen classes in our data. Their statistics and the two band (3,6) ellipse plot are shown in Figures 5 and 6 respectively.

We decided to use the detailed analysis results performed by the LARS professional analyst to do the estimation. The coding parameter estimation



Chan. 6 .6-.7μM



Chan. 8 .8-1.1μM

LEGEND

A =	CLASS	1	HIGHWAY
B =	CLASS	2	AG2-1
C =	CLASS	3	AG2-2
D =	CLASS	4	AG2-3
E =	CLASS	5	AG2-4
F =	CLASS	6	AG2-6
G =	CLASS	7	AG2-7
H =	CLASS	8	PAST-WHT
I =	CLASS	9	PAST
J =	CLASS	10	PAST2-11
K =	CLASS	11	FOR2-12
L =	CLASS	12	FOR2-13
M =	CLASS	13	COM1-1
N =	CLASS	14	RES1-3
O =	CLASS	15	RES1-7
P =	CLASS	16	DARK AG
Q =	CLASS	17	WATER

Figure 5. Cospectral Plot.

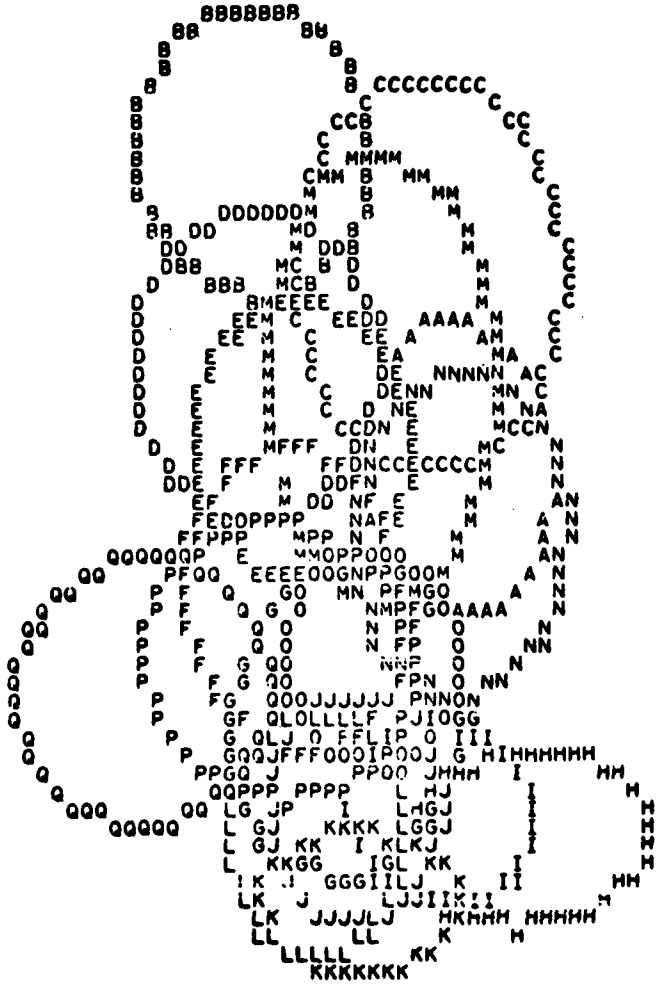


Figure 6. Ellipse Plot of Channel 3 Verses 6

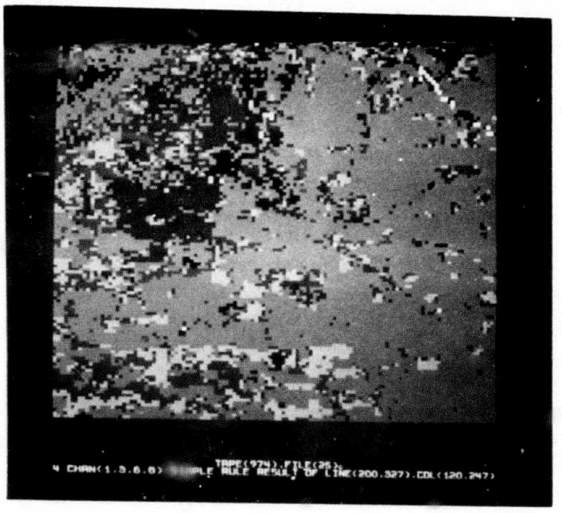
result and the evaluated variances and covariances are given in Table 1.

Band	$\hat{a}$	V_{00}	V_{01}	V_{11}	V_{02}
0.7-0.8 μM	0.228	1.489	0.5372	0.3216	0.221113
0.6-0.7 μM	0.229	1.486	0.5370	0.3212	0.2210

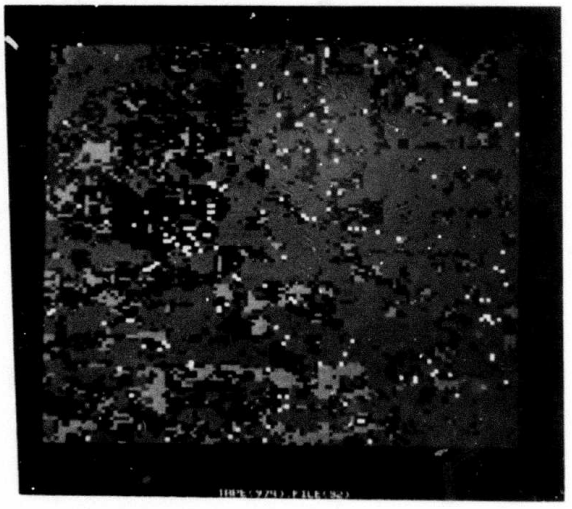
Table 1. Coding estimation result.

The statistics for the 17 classes were obtained by the analyst at LARS with the aid of reference data such as aerial photography, U.S. geographic maps etc. However, it is extremely hard to use this reference data to obtain performance accuracy because, for one thing, it is hard to store point-by-point ground truth in the computer. Therefore we decided to examine our results visually from the positive-negative films.

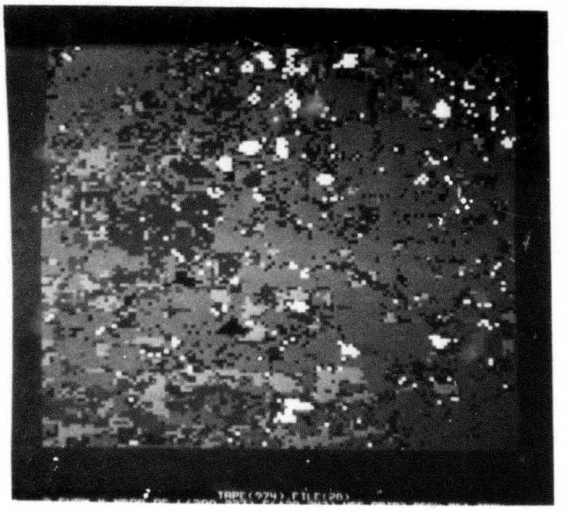
Two areas each of size 128 x 128 were used for testing the contextual algorithm performance. These results are shown in Figure 7,8 for block 1 (line 200-327, col. 120-247) and block 2 (line 73-200, col. 90-217) respectively. The results are presented with the two-band and 4-band simple rule results. It can be seen that residential areas (to the left of the picture in Fig. 7), forests and pastures are all classified equally well for the three results. However, highway and commercial areas (white dots) are detected much better in the contextual rule results while just barely detected in the no context 4-band simple rule results. To further demonstrate this, Fig. 9 presents results of classes for water (river), highway, and commercial areas as they are all represented by white color dots. It is not hard to see that, with the help of context information, these seldom occurring classes (represented by low a priori probabilities) are correctly



4 band simple result

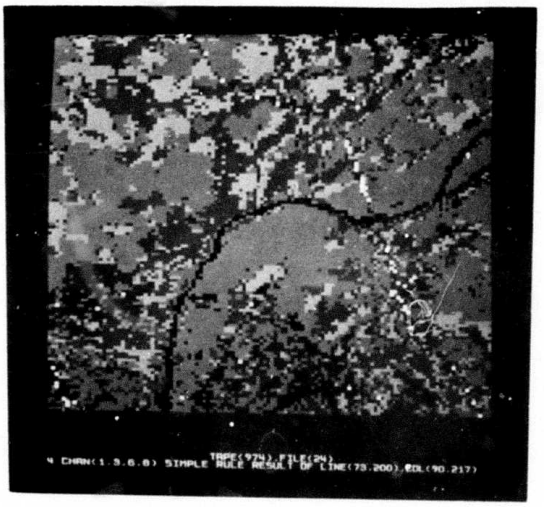


4 band simple result (2:37 Min.)

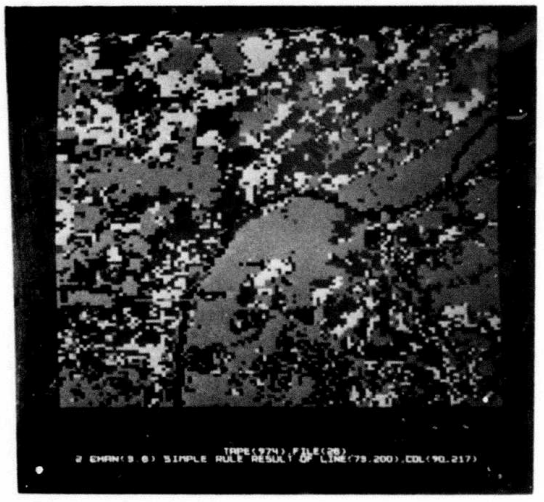


2 band context result (9:18 Min)

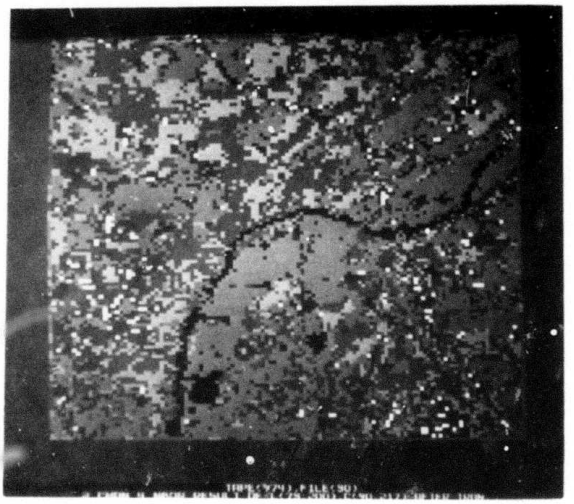
Fig. 7 Experimental Result for Block 1



4 band simple result

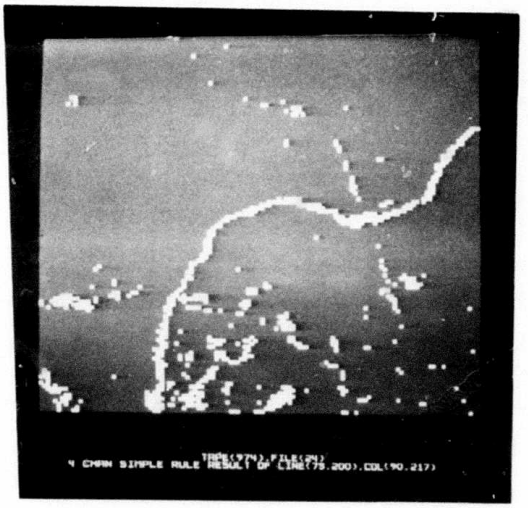


2 band simple result (2:37Min)

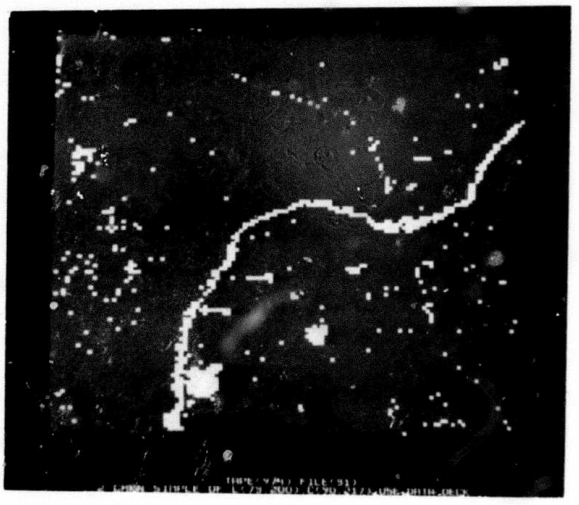


2 band context result (9:18 Min)

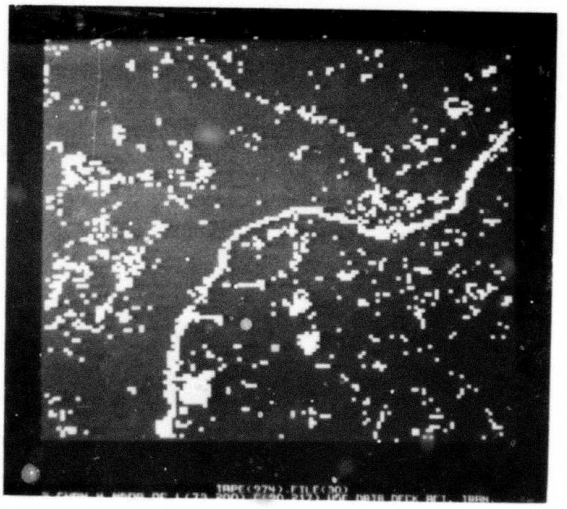
Fig. 8 Experimental Result for Block 2



4 band simple result



2 band simple result



2 band context result

Fig. 9

classified. The estimated performance improvement in both block areas is about 5%.

The computation time for the 4-neighbor (NBOR) context rule is less than for the 4-band simple rule while the performance is better.

*LARS, Laboratory for Applications of Remote Sensing, Purdue University, West Lafayette, Indiana 47907.

Acknowledgement

This work was supported by the Advanced Project Research Agency Grant MDA903-77-G-1.

References

1. T. S. Yu and K. S. Fu, "Contextual Pattern Classification of Remotely Sensed Multispectral Data", Presented at the 8th Annual Image Modeling and Simulation Conference, Univ. of Pittsburgh, 1977.

2. J. R. Welch and K. G. Salter, "A Context Algorithm for Pattern Recognition and Image Interpretation", IEEE Trans. SMC, vol. SMC-1, pp. 24-30, Jan. 1971.

3. Besag, J. E. "Spatial Interaction and the Statistical Analysis of Lattice Systems", J. R. Stat. Soc. B 36, 192-236, 1974.

4. Moran, P. A. P. "A Gaussian Markovian Processes on a Lattice", J. Appl. Prob., 10, 54-62, 1973.

5. Abramowitz, M. and Stegum, I. A., Handbook of Mathematical Functions, Dover, New York, 1965.

6. Besag, J. E. and Moran, P. A. P., "On the Estimation and Testing of Spatial Interaction in Gaussian Lattice Processes," Biometrika, pp. 555-562, 1975.

7. K. Fukunaga, "Introduction to Statistical Pattern Recognition," Academic Press, 1972.

8. Besag, J. E., "Nearest Neighbor System and the Auto-Logistic Model for Binary Data," J. R. Stat. Soc. B, 34, 75-83, 1972.

9. K. S. Fu, "Pattern Recognition in Remote Sensing of Earth Resources," IEEE Trans. on Geo. Scien. & Elec., Jan. 1976.

10. Moran, P. A. P., "Necessary Conditions for Markovian Processes on a Lattice," J. Appl. Prob., 10, 605-612, 1973.

STABILITY OF GENERAL TWO-DIMENSIONAL RECURSIVE FILTERS

B. O'Connor and T. S. Huang

INTRODUCTION

Two-dimensional recursive digital filters have generated much interest lately. They have the potential of saving computer time and memory. After defining recursive filters from a different point of view we will present a number of stability theorems. Here, it will be shown that any general recursive filter can be mapped into a first quadrant filter. Furthermore, a theorem will be developed which relates the stability of any digital filter to its two-dimensional phase function. Finally, a number of practical stability tests will be presented including one which consists of checking the root distribution of one one-dimensional polynomial.

GENERAL RECURSIVE FILTERS

A general recursive filter can be described by the following recursive equation:

$$O(m,n) = \sum_{(r,s)\in\alpha} a(r,s)\, I(m-r,n-s) - \sum_{(k,\ell)\in\beta-(0,0)} b(k,\ell)\, O(m-k,n-\ell) \qquad (1)$$

where $a(r,s)$ and $b(k,\ell)$ are real finite extent arrays with respective lattice supports of α and β and $I(m,n)$ and $O(m,n)$ are the respective input and output arrays. Furthermore, we will assume that $b(0,0) = 1$ and that $\beta = \{(m,n) : b(m,n) \neq 0\}$ is contained in a lattice sector with center $(0,0)$ of angle less than π. These conditions guarantee that for a large class of inputs equation (1) can be solved by incrementing the values of the indices (m,n) in such a fashion that all values of the output can be computed in turn from a given set of initial conditions [1,2]. In other words, the stated conditions imply that equation (1) is recursible for many inputs.

It is our desire to use equation (1) to implement a linear shift-invariant (LSI) system on the input $I(m,n)$. Therefore, all initial conditions

must be zero. The impulse response of this LSI system is defined by the following recursive equation:

$$h(m,n) = a(m,n) - \sum_{(r,s)\epsilon\beta-(0,0)} b(r,s)\, h(m-r,n-s) \qquad (2)$$

In general, the impulse response will be nonzero in some shifted lattice sector of angle less than Π. In much of the literature [3,4,5,6,7] only first-quadrant arrays a(m,n) and b(m,n) are considered which is just a particular case of the above formulation.

The following theorem summarizes and extends the above discussion.

Theorem 1: Let a(m,n) and b(m,n) be two real finite extent arrays satisifying (a) b(0,0) = 1, and (b) there exists a (m,n) $\neq$ (0,0) such that b(m,n) $\neq$ 0. Then equation (1) is recursible if and only if (iff)

(i) $\beta = \{ (m,n) : b(m,n) \neq 0 \}$ is a subset of a lattice sector with center (0,0) of angle less than Π.

(ii) i(m-r,n-s) as a function of (r,s) does not intersect β^* (which is the minimum angle lattice sector containing β) at an infinite number of points for all (m,n).

If b(m,n) satisfies conditions (a), (b), and (i) of the above theorem, then it will be called a recursive filter array. Associated with a recursive filter array is a lattice sector called β^* which consists of all lattice points in the minimum angle sector containing β. β^* can be uniquely defined by two vectors

$$\beta_1 = (M_1,N_1) \epsilon Z^2 \text{ and } \beta_2 = (M_2,N_2) \epsilon Z^2$$

as $\beta^* = S[(M_1,N_1), (M_2,N_2)]$

$= \{ (m,n) \epsilon Z^2 : (m,n) = r_1\beta_1 + r_2\beta_2,\ r_i \text{ are non-negative real numbers for } i = 1, 2\}$

where Z is the set of integers and M_1 and N_1 along with M_2 and N_2 are mutually prime integers.

For simplicity, we shall consider recursive filters with $a(m,n) = \delta(m,n)$ (unit pulse). Equations (1) and (2) now become

$$O(m,n) = i(m,n) - \sum_{(r,s)\epsilon\beta-(0,0)} b(r,s)\, O(m-r,n-s) \tag{3}$$

$$h(m,n) = \delta(m,n) - \sum_{(r,s)\epsilon\beta-(0,0)} b(r,s)\, h(m-r,n-s) \tag{4}$$

There exists another method of finding the recursive solution of equation (4) which will prove useful in the forthcoming stability discussion. Define $B(w,z)$ for a recursive filter array $b(m,n)$ as

$$B(w,z) = \sum_{(r,s)\epsilon\beta} b(r,s)\, w^r z^s. \tag{5}$$

This equation can be rewritten as $B(w,z) = 1-C(w,z)$ where $C(w,z)$ is a polynomial with no constant term. A formal expansion for $1/B(w,z)$ can be obtain-

$$1/B(w,z) = \frac{1}{1-C(w,z)} = \sum_{n=0}^{\infty} [C(w,z)]^n = \sum h(m,n) w^m z^n \tag{6}$$

We have proven in reference [8] that this sequence $h(m,n)$ is the recursive solution of equation (4), that is, the impulse response of the system. The following theorem summarizes our findings:

Theorem 2: Assume $b(m,n)$ is a recursive filter array, $h_1(m,n)$ is the solution of equation (4) and that $h_2(m,n)$ is the sequence gotten from equation (6); then $h_1(m,n) = h_2(m,n)$.

STABILITY THEOREMS

In most applications only bounded-input bounded-output (BIBO) stable filters are of interest. For a LSI system a necessary and sufficient condition for BIBO stability is that $\sum |h(m,n)| < \infty$. Therefore, the recursive filter represented by the recursive filter array $b(m,n)$ is stable if and only if the impulse response obtained from equation (4) is absolutely summable,

that is, $\sum_{(m,n)\in\beta^*} |h(m,n)| < \infty$. If this impulse response is transformed into another impulse response by a mapping of the form

$$\begin{aligned} m' &= k_1 m + k_2 n \\ n' &= k_3 m + k_4 n \end{aligned} \tag{7}$$

where $k_i \in Z$ then it is still absolutely summable. Furthermore, if $k_1 \cdot k_4 - k_2 \cdot k_3 \neq 0$, the resulting sequence is just the original sequence reordered and thus if it is stable then the original impulse response is stable. Moreover, the resulting impulse response corresponds to the mapped original recursive filter array by $g(m,n) = b(k_1 m + k_2 n,\ k_3 m + k_4 n)$. This fact follows easily from equation (6). These observations are formalized in the following theorem [8].

Theorem 3: $b(m,n)$ is a stable recursive filter array iff $g(m,n)$ is a stable recursive filter array where $g(m,n) = b(k_1 m + k_2 n,\ k_3 m + k_4 n)$ where $k_1 k_4 - k_2 k_3 \neq 0$ and $k_i \in Z$.

This theorem is important because it allows any class of filters to be mapped into any other class while at the same time preserving stability. In particular, if $b(m,n)$ is a recursive filter array with $\beta^* = S[(M_1,N_1),\ (M_2,N_2)]$ and $D = M_1 N_2 - N_1 M_2 \neq 0$ then the following mapping will change $b(m,n)$ into a first-quadrant filter

$$\begin{aligned} k_1 &= \text{sgn}\ (D)\ N_2 \\ k_2 &= -\text{sgn}\ (D)\ M_2 \\ k_3 &= -\text{sgn}\ (D)\ N_1 \\ k_4 &= \text{sgn}\ (D)\ M_1 \end{aligned} \tag{8}$$

Therefore, the stability of a general recursive filter can be determined by testing the stability of the first-quadrant filter obtained from equation (8).

In the literature, there exist a quarter-plane stability testing procedures [3,4,5,6,7]. For example, any of these methods can be applied to G(W,Z) in the below example.

Example: $B(w,z) = .5\, w^{-1}z + 1 + .85w + .1wz + .5w^2z^{-1}$ $\beta^* = S[(2,-1), (-1,1)]$. Equation (11) yields $w = WZ$, $z = WZ^2$ to get $G(W,Z) = 1 + .5W + .5Z + .85WZ + .1W^2Z^3$. Now g(m,n) can be shown to be stable; therefore, b(m,n) is stable.

Most of the stability theorems of first-quadrant filters relate stability to the zero distribution of $B(w,z) = \sum_{(m,n)\in\beta} b(m,n)\, w^m z^n$. The following theorem summarizes these results.

Theorem 4: Let b(m,n) be a first-quadrant recursive filter array then it is stable iff

(1) $B(w,z) \neq 0 \quad |w| \leq 1, |z| \leq 1$ [3,4]

iff

(2) (a) $B(w,z) \neq 0 \quad |w| = 1, |z| \leq 1$ [3,9]

(b) $B(w,b) \neq 0 \quad |w| \leq 1$ b a constant $|b| \leq 1$

iff

(3) (a) $B(w,z) \neq 0$ on $T^2 = \{ (w,z) : |w| = 1, |z| = 1\}$ [7]

(b) $B(a,z) \neq 0 \quad |z| \leq 1 \quad |a| = 1 (|a| \leq 1)$

(c) $B(w,b) \neq 0 \quad |w| \leq 1 \quad |b| \leq 1 (|b| = 1)$

iff

(4) (a) $B(w,z) \neq 0$ on T^2 [7]

(b) $B(z,z) \neq 0$ on $|z| \leq 1$

iff

(5) (a) $B(w,z) \neq 0$ on T^2 [8,10]

(b) $B(z^{\ell_1}, z^{\ell_2}) \neq 0 \; |z| \leq 1$ for any $\ell_1, \ell_2 \in Z^+$.

Utilizing parts of this theorem along with our mapping theorem we can obtain a stability theorem for general recursive filters.

Theorem 5: $b(m,n)$ is a stable recursive filter array iff

(a) $B(w,z) \neq 0$ on T^2

(b) (i) if $D = M_1N_2 - N_1M_2 \neq 0$

$B(\lambda^{p1}\ \lambda^{p2}) \neq 0$ for $|\lambda| \leq 1$

for any p_1 and p_2 where

$p_1 = \text{sgn}\ (D)\ N_2\ \ell_1 - \text{sgn}\ (D)\ N_1\ \ell_2$

$p_2 = -\text{sgn}\ (D)\ M_2\ \ell_1 + \text{sgn}\ (D)\ M_1\ \ell_2$

and $\ell_i \in Z^+$

(ii) if $D = 0$

$B(\lambda^{p1}, \lambda^{p2}) \neq 0$ for $|\lambda| \leq 1$

and $p_1 = M_1\ \ell_1$, $p_2 = N_1\ \ell_2$

Theorems (4) and (5) along with some mathematical results from Rudin [10] can be utilized to prove the following theorem.

Theorem 6: Let $b(m,n)$ be a recursive filter array and let $B(w,z) = \sum_{(m,n)\in\beta} b(m,n)\ w^m\ z^n$ then the LSI system $b(m,n)$ represents is stable iff

(a) $B(w,z) \neq 0$ on T^2

(b) $B(w,1)$ and $B(1,z)$ have no linear phase components

By a polynomial in z and z^{-1} having no linear phase components we mean that it has no roots on unit circle and that the phase function associated with the Fourier transform of the sequence have no linear term.

The conditions of this theorem are equivalent to those in part (3) of Theorem 4 for first-quadrant filters [7]. However, this theorem is much more general because it is true for any type of filter. Hence, no longer is it necessary to check different zero region conditions of $B(w,z)$ for each different type of filter as done in the past [3-7].

Conditions (a) and (b) can be restated respectively as (i) the Fourier transform of b(m,n) is never equal to zero, and (ii) there exist no linear phase components in the phase function. Dudgeon [11] has shown that if conditions (i) and (ii) hold then the phase of the Fourier transform is continuous, odd, and periodic. The above observations suggest the following theorem:

Theorem 7: A recursive filter represented by a recursive filter array is stable iff the unwrapped phase is continuous, odd, and periodic.

This theorem is important because it relates the phase of a two-dimensional recursive filter to its stability. This will aid in our development of efficient, practical stability testing algorithms.

PRACTICAL STABILITY TESTS

In this section several efficient stability algorithms for general recursive filters will be considered. These algorithms test the validity of the conditions given in either Theorem (5) or (6). First, consider condition (b) of both theorems. Here, a one-dimensional polynomial's root distribution with respect to the unit circle must be determined. This information can be obtained by unwrapping the phase of sequences which form these polynomials. A very efficient and accurate phase unwrapping method has been recently developed [12]. However, it has been our observation that for a polynomial of order less than two-hundred or so, analytic methods such as the Marden-Jury table [13,14] test can obtain this information more efficiently.

If condition (b) of either theorem is satisfied then B(w,z) must be checked for zeros on T^2. If there exists a point where the Fourier transform is zero this implies that the phase is discontinuous at that point. Now, in most cases the existence of such a discontinuity can be determined without actually finding this point. Consider either for u, v real numbers

$$B_u(z) = \sum_n \left(\sum_m b(m,n) \exp(-jum) \right) z^n \quad \text{(a)}$$

or (9)

$$B_v(w) = \sum_m \left(\sum_n b(m,n) \exp(-jvn) \right) w^m \quad \text{(b)}$$

If there exists a $(w_o,z_o) = (\exp(ju_o), \exp(jv_o))$ with $B(w_o,z_o) = 0$ then usually both $B_u(z)$ and $B_v(w)$ have root distributions with respect to unit circle which vary as a function of u and v when viewed as polynomials in z and w respectively near u_o and v_o. Thus if $0 \leq u \leq \pi$ (or $0 \leq \quad \leq \pi$) is sampled fine enough so that these discontinuities can be resolved then the existence of the point (w_o,z_o) is substantiated although the values of (w_o,z_o) are not explicitly known.

The above discussion suggests the following practical test for zeros on T^2. First, sample either $0 \leq u \leq \pi$ or $0 \leq v \leq \pi$ or both on a fine grid. The intersample distance will determine the accuracy of the test. Next, evaluate the coefficients of $B_u(z)$ or $B_v(w)$ for all grid samples. This can be done quite efficiently by employing the FFT algorithm on the rows of $b(m,n)$ to obtain the coefficients of $B_u(z)$ for $u_i = {}^{2\pi i}/_{2^N}$ or on the columns of $b(m,n)$ to obtain the coefficients of $B_v(w)$. Next, some method is used to determine the root distribution of each $B_{u_i}(z)$ or $B_{v_i}(w)$ with respect to the unit circle. For virtually all digital filter applications these polynomials will have orders less than a hundred so the Marden-Jury table test should be one of the most efficient methods for determining this. If, however, the Nyquist plot of each polynomial via Tribolet's phase unwrapping algorithm is employed to obtain this information then this procedure becomes a very efficient implementation and generalization of DeCarlo's Nyquist-like stability test [15,16]. The last step of the procedure is to check whether this zero distribution ever changes as a function of u or v; if it does vary, the filter is unstable;

If it doesn't change, the filter may or may not be stable. However, if a very dense sampling grid was used then we can be "almost sure" that the filter is stable.

These tests can be re-interpretted in terms of the two-dimensional phase function of b(m,n). The above methods determine whether the phase contains a linear phase component along a series of either vertical (9a) or horizontal (9b) lines [See figure 1(a) and (b)]. This interpretation suggests other procedures which are also motivated by Mersereau's One-Projection Theorem and the Projection-Slice Theorem [17,18]. Instead of approximately parameterizing the u-v plane by horizontal or vertical lines we can use slanted lines as depicted in Fig. 1(c) and 1(d). These correspond to projections of angle $\tan^{-1}(1/N)$ or $\tan^{-1}(1/M)$ where M and N are the length and width of the zero-padded b(m,n) array. These projections can be obtained by operating on the one-dimensional arrays formed by concatenating the zero-padded columns and zero-padded rows respectively.

If these one-dimensional polynomials contain a linear phase component the filter is definitely unstable. The accuracy of this test can be increased by adding more zeros to the original array's columns or rows. Therefore, for large enough M or N we can be "almost sure" that a given filter is stable.

Figures 2 and 3 summarize both the column and rows tests and the concatenation tests.

EXAMPLES

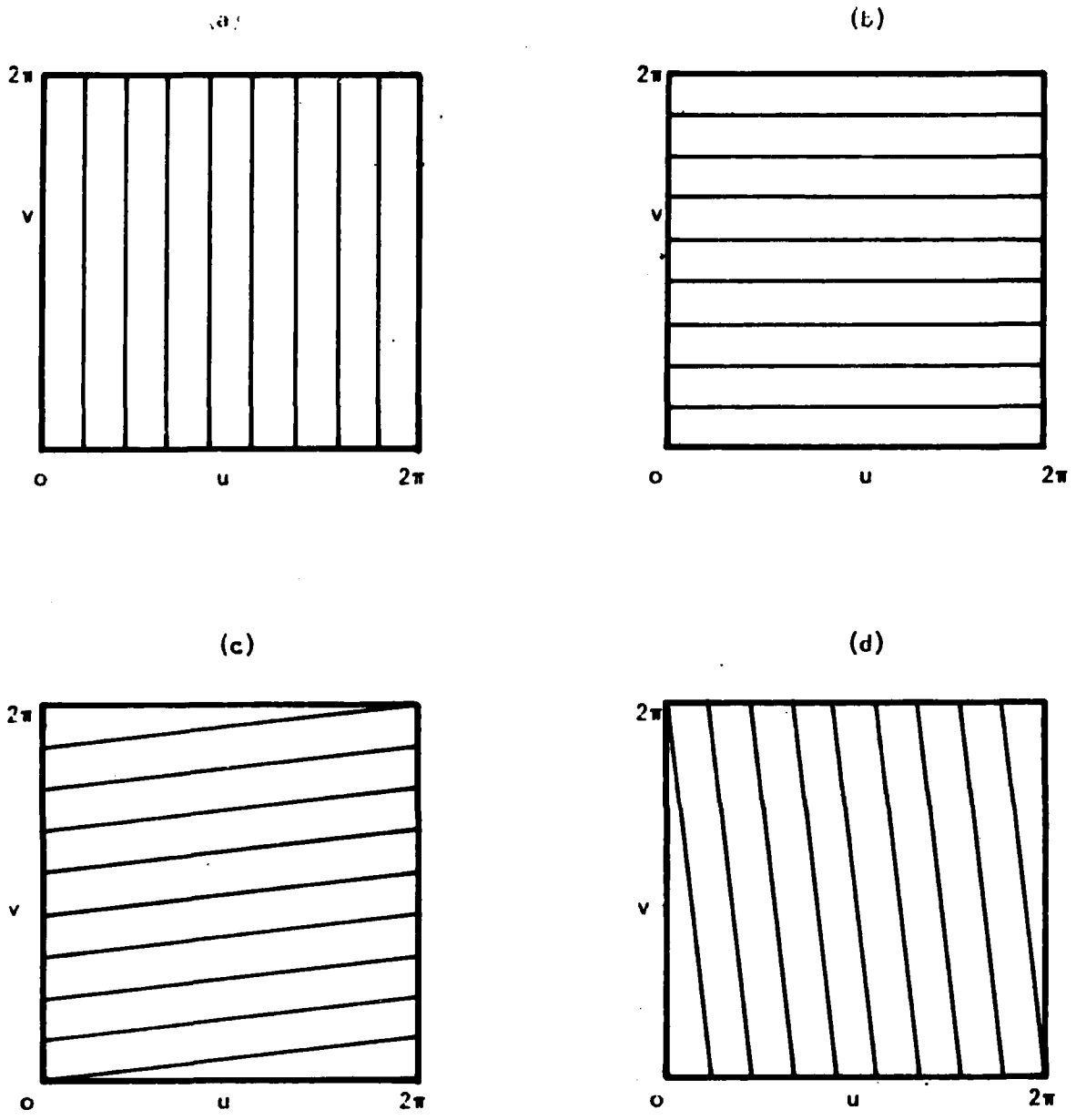
Example: $B_1(w,z) = .5w^{-1}z + 1 + .85w + .1wz + .5w^2z^{-1}$

Condition 7(b) :

$B_1(w,1) = .5w^{-1} + 1 + .95w + .5w^2$

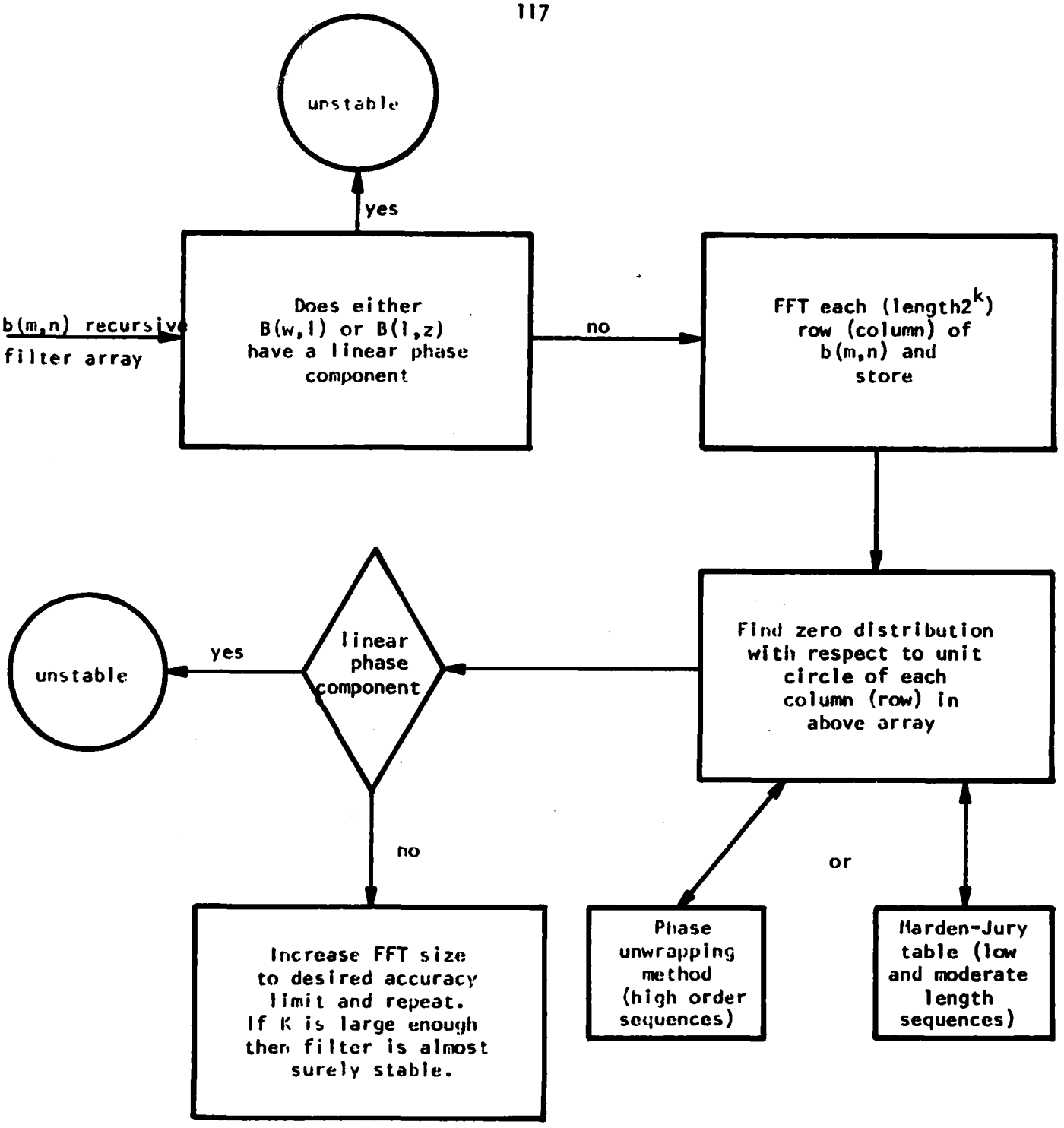
$B_1(1,z) = .5z^{-1} + 1.85 + .6z$

Both have no linear phase components



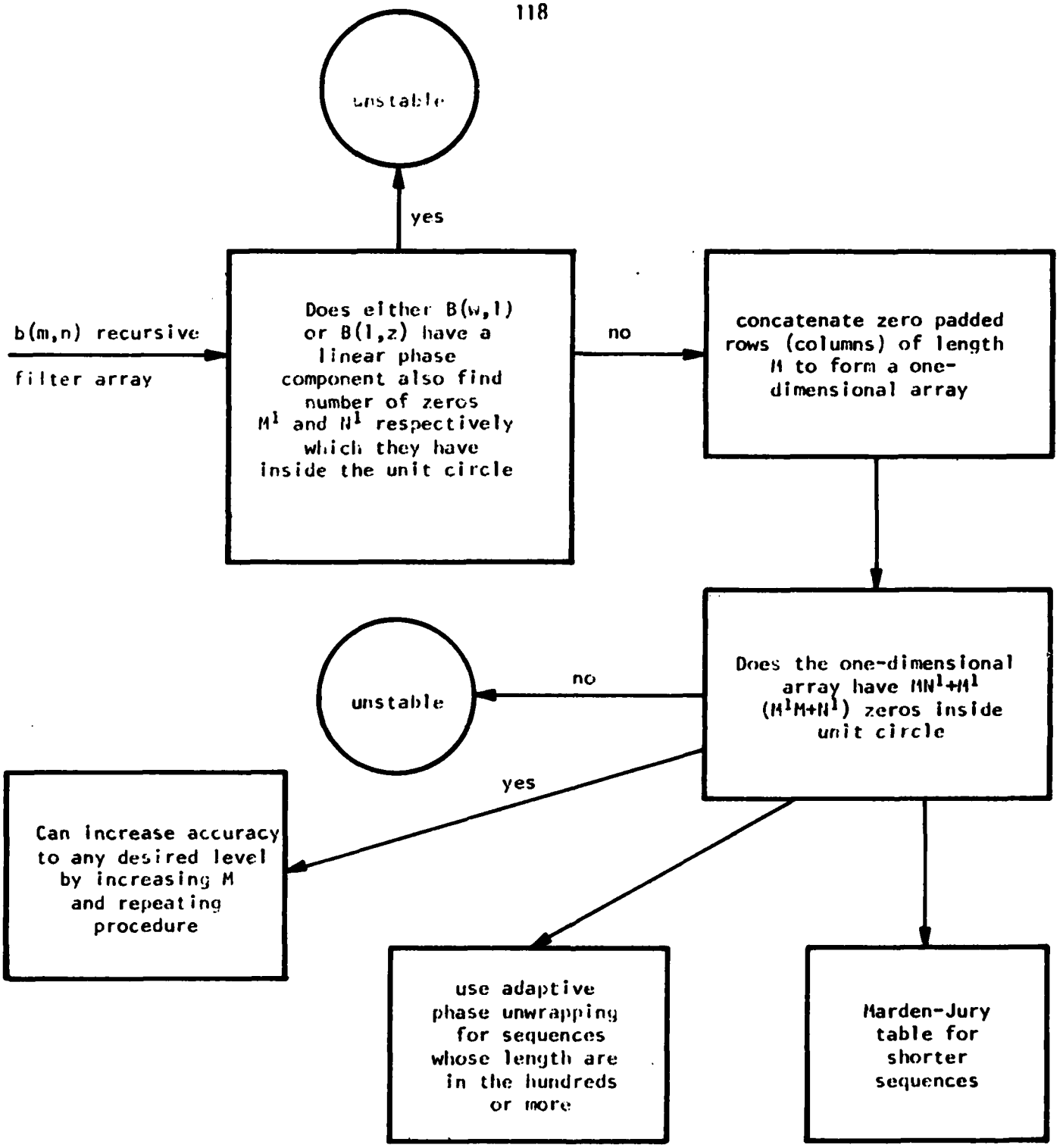
Parameterization of torus T^2 using different phase unwrapping methods.

FIGURE 1



Efficient stability algorithms for two-dimensional recursive filters (row and column algorithms).

FIGURE 2



Stability algorithms for two-dimensional filters which require only one one-dimensional root distribution test (concatenation algorithms).

FIGURE 3

Condition 7(a): [row test]

Each row was transformed via a 1024 length FFT and the root distribution of the resulting columns revealed that, at least for this accuracy, $B_1(w,z) \neq 0$ on T^2. (Since b(m,n) is real only approximately half the number of root distribution tests are needed). Therefore, b(m,n) is "most likely" a stable filter array.

Condition 7(a): [concatenation test]

(i) $B_1(z,z^{-N}) = .5z^{-N-1} + .1z^{-N+1} + 1 + .85z + .5z^{N+1}$

(ii) $B_1(w^M,w^{-1}) = .5w^{-M-1} + 1 + .1w^{M-1} + .85w^M + .5w^{2M+2}$

For these to have no linear phase terms the number of roots inside the unit circle for (i) should equal N+1 and M+1 for (ii) where M and N are the lengths of the zero padded rows and columns respectively. Below is a table which lists results of using the concatenation algorithm for different values of M and N. The root distributions were determined via a modified adaptive phase unwrapping algorithm [24]. Notice that the zero distributions always corresponds to that of a stable filter.

TABLE 1

M or N Value	Length of M Sequence	Length of N Sequence	Ideal Zero Distribution Inside Unit Circle	M and N Sequence Zero Distribution
12	37	27	13	13 - 13
18	55	39	19	19 - 19
36	109	75	37	37 - 37
72	217	147	73	73 - 73
144	433	291	145	145 - 145
288	--	579	289	-- - 289

Example: $B_2(w,z) = .5w^{-1}z + 1 + .89w + .1wz + .5w^2z^{-1}$

$B_2(w,1)$ and $B_2(1,z)$ have no linear phase term. When the row algorithm is employed a phase discontinuity occurs for FFT sizes of 32 or larger. Therefore $b_2(m,n)$ is definitely unstable. Alternately this fact can be determined by using a concatenation algorithm.

Concatenations of zero padded rows and columns yield (i) $B_2(z,z^{-N}) = .5z^{-N-1} + .1z^{-N+1} + 1 + .89z + .5z^{N+2}$. (ii) $B_2(w^M,w^{-1}) = .5w^{-M-1} + 1 + .1w^{M-1} + .89w^M + .5w^{2M+1}$. If $b(m,n)$ is stable then the number of zeros inside the unit circle of the above polynomials should equal N+1 and M+1 respectively for all values of M and N. The following table indicates results for different values of M and N. A phase discontinuity appears when $M \geq 18$ and $N \geq 12$ and hence, this test also indicates that there exists a $(w_o,z_o) \in T^2$ such that $B(w_o,z_o) = 0$ so $b(m,n)$ is unstable.

TABLE 2

M or N Value	Length of M Sequence	Length of N Sequence	Ideal Zero Distribution Inside Unit Circle	M and N Sequence Zero Distribution
6	19	15	7	7 - 7
12	37	27	13	13 - 11
18	55	39	19	21 - 17
24	73	51	25	29 - 23
30	91	63	31	35 - 29
36	109	75	37	43 - 35
72	217	147	73	83 - 71
144	433	291	145	165 - 141
288	--	579	289	- 279

Example: $B_3(w,z) = 1 - 1.2w + .5w^2 - 1.5z + 1.8wz - .75w^2z + .6z^2 - .72wz^2 + .2718w^2z^2$.

The row algorithm detects a phase discontinuity for FFT sizes of 128 or larger while the column algorithm detects a discontinuity for FFT sizes of 32 or larger. The concatenation algorithms indicate a phase discontinuity for $B(z,z^N)$ for N = 18 and for $B(w^M,w)$ for M = 36. Therefore, these algorithms indicate that b(m,n) is unstable.

If Ekstrom's cepstrum test [19] is applied to this example no conclusion can be obtained for FFT sizes as large as 64x64. Hence, our algorithms are far superior both in computation time and sensitivity than the cepstrum stability test.

CONCLUDING REMARKS

We have presented recursive filtering from a somewhat different point of view. A general mapping theorem has been formulated which allows any type of filter to be mapped into a first-quadrant filter. This first-quadrant filter is stable if and only if the original filter was stable. Several general stability theorems which relate stability to the zero set of B(w,z) have been presented. These theorems led to the conclusion that a filter is stable if and only if its phase function is continuous, odd, and periodic. This observation suggested several practical stability testing algorithms. Among these are several methods which appear to be extremely efficient for high order filters. All the results in this paper can be generalized to n-dimensional filters. Moreover, the practical stability tests can be applied to any finite two-dimensional array to determine if its cepstrum exists by determining if the Fourier transform of the array ever equals zero.

REFERENCES

[1] Dudgeon, Dan E., "Two-Dimensional Recursive Filtering," Sc.D. Thesis, M.I.T. E.E. Department, Cambridge, MA, May 1974.

[2] Mersereau, Russell M. and Dudgeon, Dan E., "Two-Dimensional Digital Filtering," Proc. IEEE, vol. 63, pp. 610-623, April 1975.

[3] Huang, Thomas S., "Stability of Two-Dimensional Recursive Filters," IEEE Trans. Audio Electroacoust., vol. AU-20, pp. 158-163, June 1972.

[4] Shanks, J.L., S. Treitel, and J. H. Justice, "Stability and Synthesis of Two-Dimensional Recursive Filters," IEEE Trans. Audio Electroacoust., vol. AU-20, pp. 115-128, June 1972.

[5] Anderson, B., and E. Jury, "Stability Test for Two-Dimensional Recursive Filters," IEEE Trans. Audio Electroacoust., vol. AU-21, August 1973.

[6] Maria, G., and M. Fahmy, "On the Stability of Two-Dimensional Digital Filters," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-21, pp. 470-472, October 1973.

[7] DeCarlo, R.A., T. Murray, and R. Saeks, "Multivariable Nyquist Theory," Int. J. Control, vol. 25, pp. 657-675, 1977.

[8] O'Connor, Brian T., and Thomas S. Huang, "Stability of General Recursive Filters," Purdue University Tech. Rept., TR-EE 77-36, October 1977.

[9] Goodman, Dennis, "An Alternate Proof of Huang's Stability Theorem," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-24, pp. 427-477, October 1976.

[10] Rudin, Walter, Function Theory in Polydiscs, New York: Benjamin, 1969.

[11] Dudgeon, Dan E., "The Existence of Cepstra of Two-Dimensional Rational Polynomials," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-23, pp. 242-243, April 1975.

[12] Tribolet, J., "A New Phase Unwrapping Algorithm," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-25, pp. 170-177, April 1977.

[13] Marden, M., The Geometry of the Zeros of a Polynomial in a Complex Variable, New York: American Mathematical Society, 1949.

[14] Jury, E., Theory and Applications of the z-Transform Method, New York: Wiley, 1964.

[15] DeCarlo, Raymond, "An Algebraic Topological Approach to Stability Theory," Ph.D. Thesis, Texas Tech., EE Department, Lubbock, Texas 1976.

[16] DeCarlo, R., R. Sacks, and J. Murray, "A Nyquist-Like Test for the Stability of Two-Dimensional Digital Filters," Proc. IEEE, vol. 65, pp. 978-979, June 1977.

[17] Mersereau, Russell M., and Alan V. Oppenheim, "Digital Reconstruction of Multidimensional Signals from Their Projections," Proc. IEEE, vol. 62, pp. 1319-1338, October 1974.

[18] Mersereau, Russell M., and Dan E. Dudgeon, "The Representation of Two-Dimensional Sequences as One-Dimensional Sequences," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-22, pp. 320-325, October 1974.

[19] Ekstrom, Michael P., and John W. Woods, "Two-Dimensional Spectral Factorization with Applications to Recursive Filtering," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-24, pp. 115-128, April 1976.

SEGMENTATION OF TACTICAL TARGETS IN FLIR IMAGERY

S. G. Carlton and O. R. Mitchell

I. INTRODUCTION

We have been continuing cooperation with Honeywell System and Research Division in developing a system to detect and recognize tactical targets in FLIR imagery. Although the ultimate goal is to incorporate syntactic methods wherever appropriate and useful, this report describes statistical segmentation methods and introduces a potentially useful classification method.

Shown in Fig. 1, upper left corner, and in Fig. 3 and 8 are sample FLIR tactical targets. These images are thermal and several characteristics apply to active vehicles: (1) the motor is usually visible as a hot spot, (2) edges can be detected within and along the target, and (3) the average grey level (temperature) of the object is often different from the background. These characteristics are presently used by the Honeywell Autoscreener System to locate potential target areas.

The techniques described here assume that the images have been preprocessed by the Autoscreener and the approximate location and size of each potential target are known. We attempt a segmentation into target and nontarget pixels in the target area and proceed to look at a method for classifying the segmented targets.

II. SEGMENTATION FEATURES

To accomplish target segmentation, background statistics are collected over an annular region surrounding the potential target. Then the statistics of the target region are compared to those of the background, and points not matching the background are labeled as target points. As is evident from the sample images, grey level alone is not always enough information for accurate segmentation, so additional features are necessary for the process.

Hopefully, once the right features are selected any points that have different features than the surrounding background points will be part of the target.

The two additional features are chosen to complement the grey level image. These are texture and edges. The texture was chosen because it seems probable that object and background textures would not be identical, assuming a good texture measure were available to differentiate among textures. The edges were chosen as a feature due to the predominance of edges along the target background interface and the fact that grey level (temperature) and texture become ambiguous near the object boundaries. The edge and texture features are shown in Fig. 1 at the upper right and lower left, respectively.

The edge feature is a gradient type measurement measured over a 5x5 window for each point. The absolute difference between the upper 10 points and the lower 10 points is compared against the absolute difference between the left 10 points and the right 10 points. The center point is then replaced by the maximum of the absolute values of these two differences. This process is repeated for each point in the original image to produce the edge feature image. Figs. 4 and 9 are also edge feature images based on the original images in Figs. 3 and 8, respectively.

The texture feature is derived from the max-min local extrema described in previous reports [1-2]. Local grey level extrema are measured in hysteresis smoothed versions of the original iamge using three smoothing thresholds. The lowest level extrema correspond mostly to noise in the image, whereas the highest correspond mostly to edges. The remaining medium level extrema are a measure primarily of the texture in the image. These medium level extrema locations are shown in Fig. 2 beside the original tank image from Fig. 1. Fig. 5 shows the medium level extrema extracted from Fig. 3.

The texture feature image is created from the extrema by averaging the number of medium level extrema in every 10x10 window in the image and replacing the center point of that window with the average. Texture feature images are shown in Fig. 1 (lower left), Fig. 6, and Fig. 10.

III. SEGMENTATION PROCEDURES

Once the feature images are produced two concentric circles are centered at each potential target as derived from the Honeywell preprocessing system. The inner circle represents the potential target area and the annular region between the two circles represents the background region. In an automated system these circle sizes would be adaptive since approximate target size and background context will also be available from prior processing stages. In our implementation of the system here, the inner radius was fixed at 30 pixels and the outer radius at 60 pixels. The background annular region must be large enough to allow a sufficient background sample to be collected but it must not include target points or be so large that irrelevant background obscures the background/target differences.

The present background statistics gathering program generates a three-dimensional histogram over the original and two features for all background points. The quantization selected allows for 32 original grey levels, 8 edge values, and 16 texture values. This background histogram is therefore composed of 4096 bins.

Once the background 3-D histogram is completed, each potential target point (3-D vector) is compared against its background bin. If that feature combination occurs often in the background, the point is considered another background point. If the feature combination does not occur in the background, that point is labeled a target point.

The results of this process are shown in Fig. 1 (lower right). The target test was done over the whole image instead of just the inner circle

to give us an idea of background rejection of our process. Segmentations using this process are also shown in Figs. 7 and 11. The segmentation results using this technique are very encouraging and the potential for improvement is also high.

Present attempts at improvement of this method include a better texture measure and better statistics data. The local extrema over all smoothing thresholds contain much more texture information than is presently retained in our average over all medium extrema. We are attempting to develop an adaptive, robust technique which would measure those texture properties most evident in each image. The single background histogram might be replaced by looking at statistics over small windows of the background. This would allow the background to be represented by the statistics of only a few "average" windows. This would allow background classification as well as background/target separation.

IV. CLASSIFICATION BY PROJECTIONS

The segmentations produced by the method previously described produce results which are sometimes fragmented and contain drop-out and extraneous points. A classification scheme which is somewhat insensitive to these variations would be appropriate. We are presently investigating the use of projections through the segmented object to derive classification features. This type of structure recognition method is being developed by New Mexico State University for missile tracking at the White Sands Missile Range [3]. It has the advantage that the integration process of the projections averages out many of the noise problems inherent in thermal images and our classification method.

Shown in Fig. 12 are eight projections through a segmented object (background points set to zero, target points remain at their original grey level).

The object is a tank and is shown in the middle of Fig. 12. The small circles along the horizontal axes represent 10% area increments along the projections. The numbers printed below are the distances between the 10% area increments normalized so that the total distance (representing 64 pixels horizontally) in 1000. Note that these projections retain several characteristic features of the tank: 1) the motor hot spot; 2) the cool region between the motor and front of the tank; 3) the cannon barrel; and 4) the rectangular shape.

Shown in Fig. 13 are the projections through another tank from the same aspect angle (note the similarities). Fig. 14 shows projections through a different tank at a different angle using a different FLIR sensor. Although shape information is extractable, this target would have to be learned separately from the first two. Fig. 15 shows the projections through a segmented APC. Here the differences between an APC and a tank are most evident in the front of the objects.

The eventual goal of this projection method is to derive invariant measures (such as ratios of the 10% area increment distances) on several of the projections (such as narrowest and widest) so that objects of interest can be classified.

REFERENCES

[1] O. R. Mitchell, C. R. Myers, and W. A. Boyne, "A Max-Min Measure for Image Texture Analysis," IEEE Transactions on Computer, Vol. C-26, April 1977, pp. 408-414.

[2] S. G. Carlton and O. R. Mitchell, "Image Segmentation Using Textures and Grey Level," Proceedings of the IEEE Conference on Image Processing and Pattern Recognition, Troy, N.Y, June 6-8, 1977, pp. 387-391.

[3] G. M. Flachs, W. E. Thompson, and Yee-Hsuun U, "A Real-Time Structural Tracking Algorithm," NAECON 1976 Record, pp. 161-168.

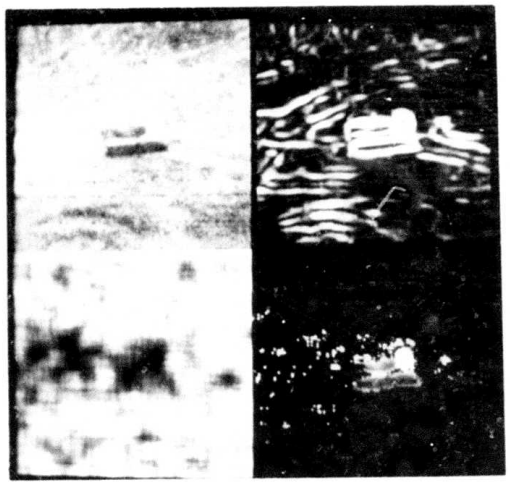


Fig. 1 Original, edges, texture, and segmented result for FLIR tank image. (each 128x128)

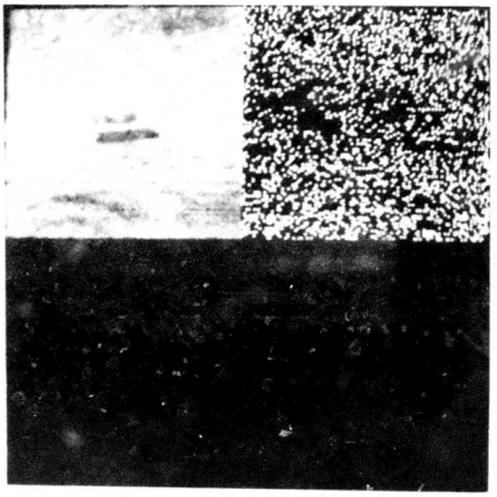


Fig. 2 Medium size local extrema used in generating texture feature

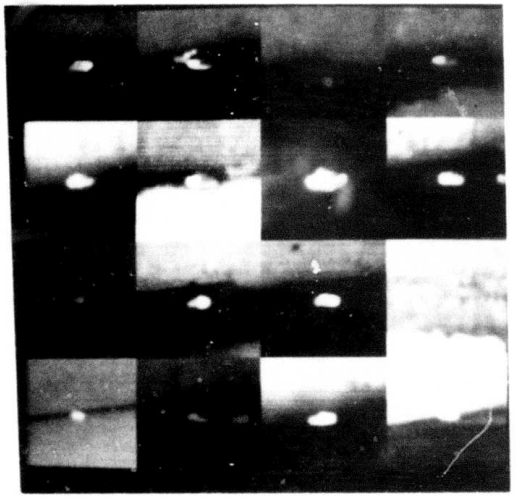


Fig. 3 Sixteen target image blocks (128x128) of NVL FLIR data. Blocks 3 and 12 do not contain targets

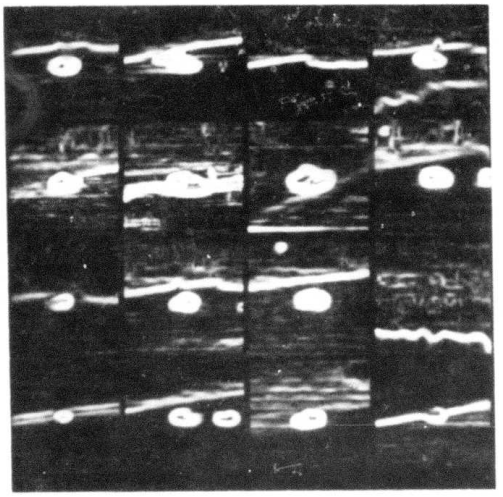


Fig. 4 Extracted edges from Fig. 3

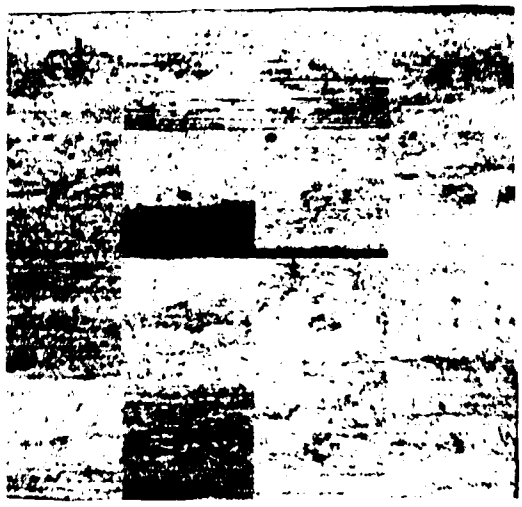


Fig. 5 Medium local extrema in Fig. 3

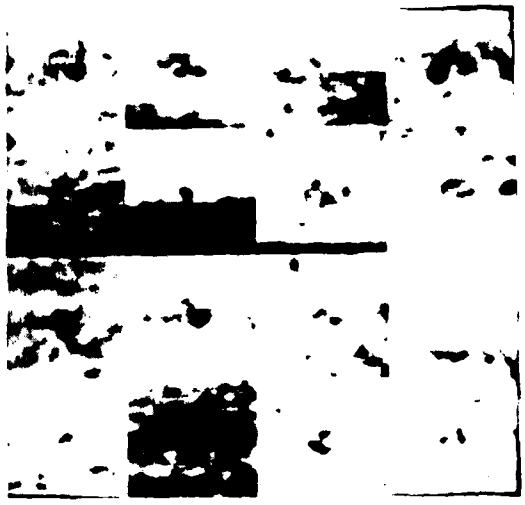


Fig. 6 Texture in Fig. 3 derived by averaging over Fig. 5.

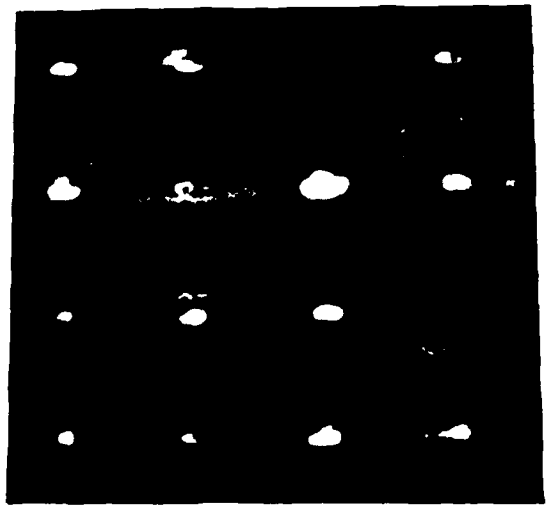


Fig. 7 Segmented targets from Fig. 3 using original, edges, and texture as three features.

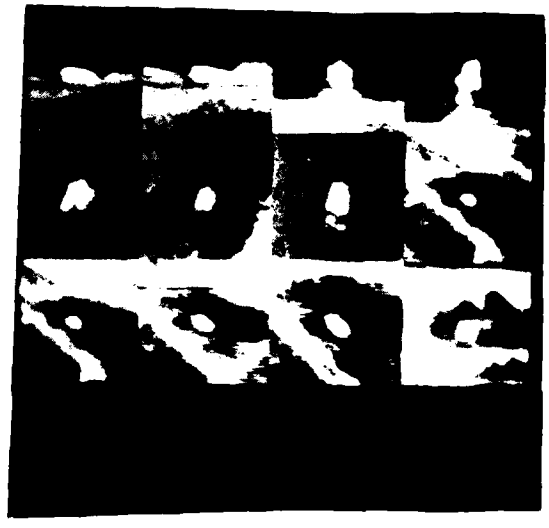


Fig. 8 Twelve targets from Honeywell FLIR data.

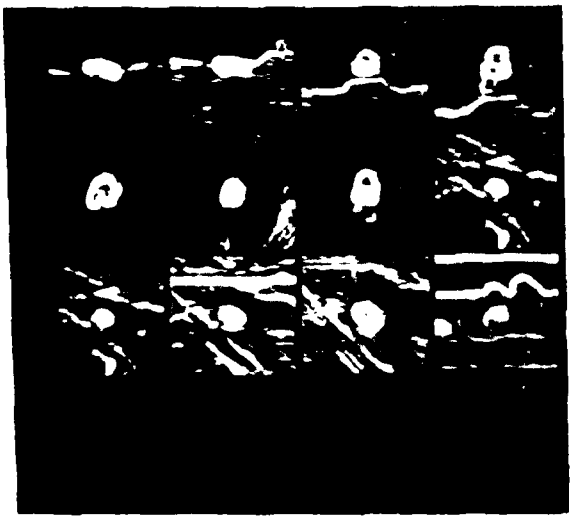


Fig. 9 Extracted edges from Fig. 8

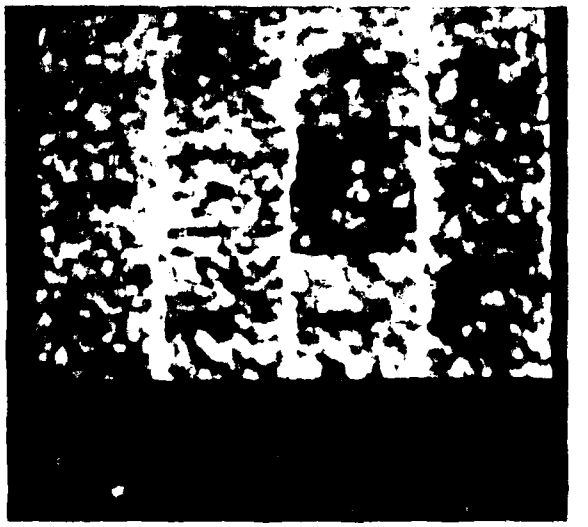


Fig. 10 Texture from Fig. 8

Fig. 11 Segmented targets from Fig. 8 using original, edges, and texture.

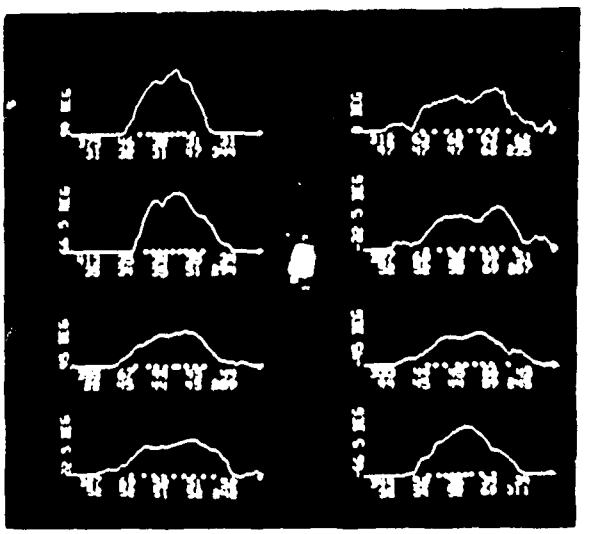


Fig. 12 Projections from segmented tank (Fig. 11, block 7)

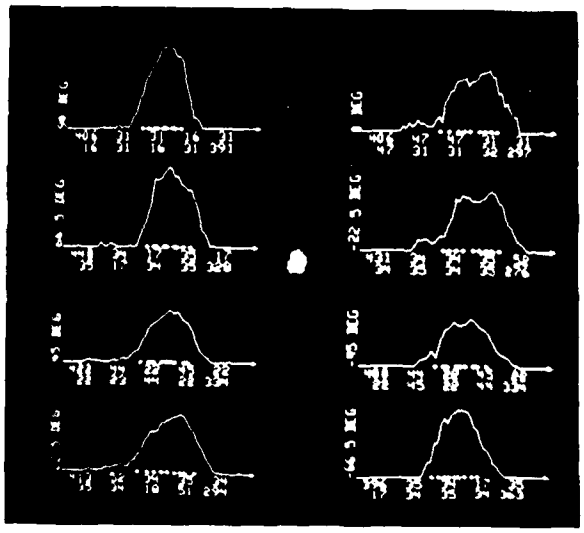


Fig. 13 Projections from segmented tank (Fig. 11, block 6)

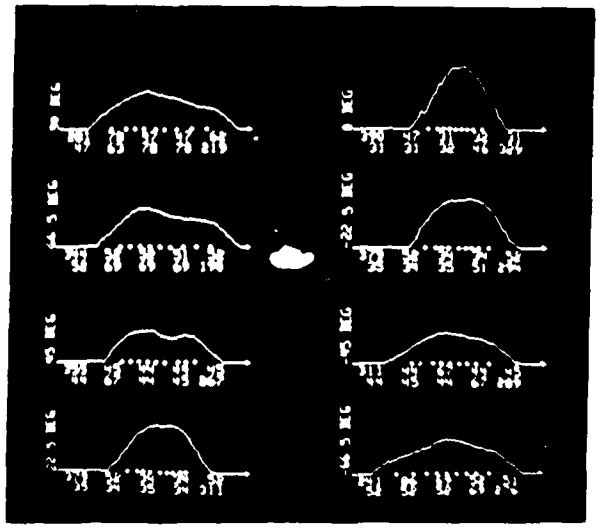


Fig. 14. Projections from segmented tank (Fig. 7, block 7)

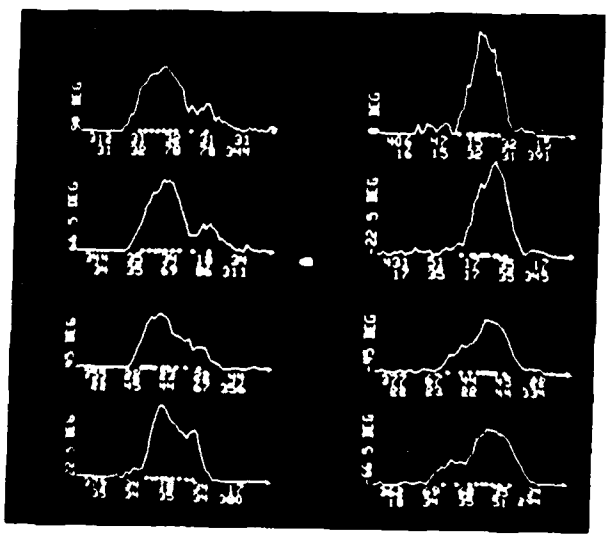


Fig. 15. Projections from segmented APC (Fig. 7, block 4)

REAL TIME TARGET TRACKING AT WHITE SANDS MISSILE RANGE

O. R. Mitchell

I. INTRODUCTION

Research was initiated this summer between Purdue and White Sands Missile Range to design segmentation and structure analysis algorithms that operate reliably on tracking imagery. In order to apply the higher level information extraction and symbol manipulation methods developed under the ARPA grant, some lower level techniques must be developed for preprocessing, target extraction, and target description. In order to accomplish these tasks a small grant has been obtained from the Army Research Office. Thus, general procedures developed under the ARPA grant may be applied to this project while a dedicated effort to the tracking problem is sustained by the ARO.

II. TRACKING SYSTEM REQUIREMENTS

Many applications require the precise location, orientation, and description of moving objects in real time. Our work at WSMR this summer has exposed us to the special requirements of some tracking problems which are best solved by optical imaging tracking systems.

Present optical tracking systems are usually "one concept" trackers, using either contrast or correlation. The contrast trackers are high speed but lack the versatility to follow a target in complex scenes. The template matching or correlation trackers require a lot of computation for reasonable window sizes and are not easily adapted to changing aspect angles, missile staging, and other common scene changes.

The need for a "smart" optical tracker exists which would operate in real-time and could adapt to target orientation and shape changes (due to staging, visible emissions, etc.), and to partial or total obscuration of the target at times.

Presently, New Mexico State University, under a grant from the Army Research Office, is developing a prototype system with some of the capabilities mentioned. However, the technical aspects of target segmentation and structure analysis need to be studied in much more detail in conjunction with the use of actual video data to develop a system that is robust under the normal variety of tracking situations encountered. We are presently doing this study.

III. DATA

We helped develop a system at WSMR this summer to put digitized video data on IBM compatible format 7-track magnetic tape. We presently have 20 digitized video images representing some tracking situations. Each image is one video field consisting of 240 by 512 points with 256 grey levels. Because the video interlace is not used, the horizontal resolution is twice that of the vertical. The target sections of sixteen fields are shown in Fig. 1. Each field has been trimmed from 240 by 512 to 64 by 128. An estimate of the motion involved can be determined from the last two targets. These images are the two fields from one TV frame, so that the changes visible have occurred in only 1/30 second.

Future research would include digitizing several continuous flights (e.g., 60 fields per second for 2 seconds) and using this data for development and testing of algorithms.

IV. RESEARCH DIRECTIONS

Our research can be divided into two major areas: (1) segmentation of potential targets from the background, and (2) structure analysis of the segmented targets.

Segmentation requires high speed processing of picture points to identify potential target areas and more detailed processing of these areas to extract a binary image of the potential target.

Structure analysis determines if the object extracted is indeed the one to be tracked, the exact location and orientation of the object, and recent changes that have occured in the object.

Both the segmentation and structure analysis sections will use information obtained from previous frames that is stored and modified in real-time to allow significant changes in the tracked object during a mission.

The segmentation of the target from the background is the most critical timing problem for real-time implementation. At video rates, a new picture element (pixel) arrives approximately every 100 nanoseconds. Thus either all initial operations must be simple or the area to be processed must be reduced by prediciting the target location in the iamge before processing. It is our opinion that the system would operate more accurately if the entire picture is processed every frame. This reduces the severe requirement of accurate prediction of target location in the picture and allows the system to acquire and discriminate interfering objects before they come confusingly close to the target being tracked.

A. Preprocessing

With the large variety of incoming im-ges, it is often necessary to do some preprocessing to make the system more invariant to changes in lighting, viewing angle, and resolution. It is also useful to enhance those aspects of the image which indicate the presence of targets (such as edges and texture of man-made objects). We are investigating the following preprocessing operations: a logarithmic input transformation, a two-dimensional median filter, a two-dimensional human visual system (HVS) filter, and averaging.

These processes are window operations that can be performed in real-time over the entire incoming video picture. The output of these processes will allow the selection of suitable potential target regions for more detailed processing.

1. Logarithmic Input Transformation

Working with the densities (log intensities) allows the system to be more invariant to illumination and lens opening changes since these produce multiplicative effects in the intensity image and additive effects in the density image. The logarithm also emphasizes the detail in the darker areas of the image which is especially useful to the texture processing algorithms described later. The logarithm is also an approximation to the visual system's action on an incoming pattern. Our research should indicate whether this non-linear operation is appropriate or necessary for typical video tracking data.

2. Two-dimensional Median Filter

The median filter allows for video dropouts including single point or single line mistakes in the analog video chain or the digitizer. A simple median filter replaces the center point of each 3x3 window with the median of all 9 points in the window. This non-linear process eliminates high frequency noise but preserves monotonic edges precisely. Shown in Fig. 2 (upper left corner) is the digitized video of a cruise missile with the contrast greatly enhanced. The results of processing this with a 3x3 median filter is shown in the upper right of Fig. 2. Notice that the edges are preserved but the noise is reduced and the data seems more correlated. The results of averaging the median filtered picture over a 3x5 window are shown in the lower left, and in the lower right is the result of averaging the original. Shown in Fig. 3 are the thresholded versions of Fig. 2. It appears that the median filter-averager combination works best for this data (the median filter removes bad points before they can be smeared by the averager).

3. Two-dimensional Human Visual System Filter

It is well known that the human visual system (HVS) responds to incoming images with an approximate logarithmic transformation followed by a spatial

and temporal bandpass filtering operation. We are investigating the usefulness of such a spatial bandpass filter in extracting objects for tracking.

A suitable filter function which is to be convolved with an input image is shown in Table 1.

Table 1. Two-dimensional HVS Filter Function

0	0	0	0	-.25	0	0	0	0
0	0	0	-.25	-.5	-.25	0	0	0
0	0	-.25	-.25	+.25	-.25	-.25	0	0
0	-.25	-.25	+.5	+1.0	+.5	-.25	-.25	0
-.25	-.5	+.25	+1.0	+1.5	+1.0	+.25	-.5	-.25
0	-.25	-.25	+.5	+1.0	+.5	-.25	-.25	0
0	0	-.25	-.25	+.25	-.25	-.25	0	0
0	0	0	-.25	-.5	-.25	0	0	0
0	0	0	0	-.25	0	0	0	0

The values are chosen to make computation of the fractional values easy for real-time implementation. Because our video data has twice the resolution in the horizontal direction than in the vertical (only one video field is processed at a time), the HVS filter is modified as shown in Table 2.

Table 2. Modified HVS Filter to Allow Half Resolution in Vertical Direction

0	0	0	-.25	-.75	-.25	0	0	0
0	-.25	-.5	+.25	+1.25	+.25	-.5	-.25	0
-.25	-.5	+.25	+1.0	+1.5	+1.0	+.25	-.5	-.25
0	-.25	-.5	+.25	+1.25	+.25	-.5	-.25	0
0	0	0	-.25	-.75	-.25	0	0	0

Both filters have a d.c. response 0.5 and a gain of 7.5 for a small object

which just covers the plus area. The filter shown in Table 2 was used to produce Fig. 5 using Fig. 4 as the input. Note that edges are emphasized and that the missile (above the plume) is enhanced.

Our research should indicate the appropriateness of such a filter and the parameters best suited to video tracking data.

B. Region Growing Using Window Statistics

After preprocessing the input video data, potential target regions must be extracted. Depending on the data, this task may be a simple thresholding operation or it may be a more complex extraction procedure. We choose to formulate the segmentation in terms of measuring statistics over each nxm window in the preprocessed picture and assigning each window to one of several classes.

The measurements which can be made on a nxm window range from a simple histogram of the data points to a representation in (n)(m)-dimensional vector space. Of course, the simplest representation that will give a proper segmentation is the most desirable. This selection can only be made after experience is gained by segmenting a large amount of typical tracking data.

Once a window parameter is selected, points must be clustered to give appropriate regions. This clustering must be based upon some distance measure between two window parameters. We are investigating the following window parameters, distance measures, and clustering techniques:

1. First Order Measure

We are using the histogram of the points in each window as the window parameter. The distance measure will be the Bayes error overlap between the two modified histograms:

$$d = 1 - \int \min [f(x) * h_1(x), f(x) * h_2(x)]\, dx \qquad (1)$$

where $h_1(x)$ and $h_2(x)$ are the grey level histograms over the two windows and $f(x)$ is a smoothing and normalizing function to be convolved with each histogram

The smoothing function is chosen so that the modified histogram has an area of unity and is smoothed so that the distance measured by Eq. (1) is not adversely affected by the discrete nature of the original histograms.

An example is shown in Figs. 6 and 7. Then +'s in Fig. 6 represent the histogram from window 1 and the 0's represent the histogram from window 2. The functions $p_1(x)$ and $p_2(x)$ in Fig. 7 represent the smoothed histograms. The shaded region is the integral in Eq. (1). This area is equivalent to the probability of error in estimating from which of these density function estimates comes a sample grey level.

The distance measure is now used to cluster windows that are close together to form segmented regions. Prior frame information should be useful here to provide initial cluster groupings. Windows that are more than a threshold distance away from all known clusters will be used to form new clusters.

2. Second Order Measure

In tracking situations where first order measures do not allow proper segmentation, a second order measure may be applied. We will use a modified joint histogram relating pairs of points within a window:

$$p(x,y) = f(x,y) * h(x,y) \qquad (2)$$

where $h(x,y)$ is the joint histogram of pairs of adjacent points (co-occurrence matrix) and $f(x,y)$ is a two-dimensional smoothing and normalizing function.

The co-occurrence matrix has been applied to texture analysis and has shown encouraging results. However, instead of calculating parameters on the matrix as is normally done, we will use the matrix itself in a two-dimensional version of the overlap error measurement of distance,

$$d = 1 - \iint \min [p_1(x,y), p_2(x,y)] \, dxdy \qquad (3)$$

Higher order measures are probably impractical but could be investigated if these initial measurements show obvious shortcomings.

C. Local Extrema Information

The detection of local grey level extrema in an image has shown to be useful in texture classification and image segmentation [1]. The detection algorithm for this operation is as follows:

In a nxm window, compare the grey level of the center point with those of its two vertical neighbors. If it is above noth neighbors, the center point is a local maximum in the vertical direction. If this is the case, compare the center point with each point along each vertical direction unit a grey level is encountered which is above the center point's value or until the edge of the window is encountered. The alrgest differences between grey levels in each vertical direction are then compared and the smallest of the two is retained as the size of the local maximum in the vertical direction. An example is shown in Table 3 for a 5x7 window.

Table 3. Sample Grey Levels for Extrema Detection

36	40	47	30	24
33	34	30	32	36
36	40	32	40	30
42	46	45	43	35
36	40	33	47	32
34	42	50	42	40
30	30	20	45	36

The center value is 45. The 47 ends the search in the top direction and the 50 ends the search in the bottom direction. The range is 15 above and 12 below. Therefore the center point is a local maximum in the vertical direction of size 12. If the point is a local minimum instead of a maximum, the process is done in the same way interchanging the above and below comparison tests.

This process is also done in the horizontal direction. In the example of Table 3, the center point is not a local extreme in the horizontal direction. If a point is a local extreme in both horizontal and vertical directions, only the largest of the two is retianed at that location. The extrema detection process is equivalent to local maximum and minimum determination following hysteresis smoothing of various amounts.

Figure 8 shows the output of such an operation on the image in Fig. 5. Extreme size is indicated by the grey level. No distinciton is made in the figure between maxima and minima or between horizontal or vertical extrema.

Note that the edges emphasized by the HVS filter now are marked by extrema. The missile orientation can be extracted from its edge information, however, the algorithm must be high-level enough so that the missile edge is extracted and not the plume. (The missile orientation angle may not be the same as its flight direction.)

Also the texture of various regions can be characterized by the types and number of extrema present. The region characterization may be useful for background classification and for plume identification. Parameters for this measurement are extracted by counting the number of various size extrema within a window surrounding each point.

D. Edge Information

The human visual system is very sensitive to edges and lines [2]. Patterns containing edges are much more visible than those conta-ning the same signal power but lacking well-defined edges. Also, many objects to be tracked are distinguished from natural environment background by the presence of edges. It would seem appropriate that one tool which should be included in the tracker repertoire is the ability to detect the use edge information. Fig. 9 shows edge intensity information extracted from the original target in Fig. 1. A gradient operator over a 3x5 window was used.

We propose to include edge information as follows: when local extrema occur at points with large edge values, they will be considered edge extrema. These edge extrema can then be used as follows: (1) the presence of an edge will enhance the possibility of a region being listed as a potential target area, and (2) windows which contain edges will be modified so that edge points control the window shape. Consider Fig. 10 as an example. In this figure, detected edge points are marked with an "E". To perform a window operation, scanning starts at the center point and moves in the numbered directions, stopping at an edge point or at the window edge. This allows for non-rectangular windows along the boundary of targets.

E. Extraction of Binary Images

The purpose of all preprocessing and segmentation techniques described in prior sections is to produce binary images which are potential targets for tracking. Information to be used in the selection of binary images will be:

(1) segmented regions from section B,

(2) texture information from section C,

(3) edge information from section D,

(4) prior frame information.

Our initial protocol for potential target selection is shown in the flow chart in Fig. 11.

F. Structure Analysis of Binary Images

The purpose of structure analysis is to find structure differences among various targets such as airplanes, helicopters, missiles, and balloons. The structure analyzer can operate on the segmented image from the previous sections We are presently investigating the potential of several structure analysis methods and will report on these results at a later time.

REFERENCES

[1] S. G. Carlton and O. R. Mitchell, "Image Segmentation Using Texture and Grey Level," Proceedings of the IEEE Conference on Pattern Recognition and Image Processing, pp. 387-391, June 1977.

[2] T. N. Cornsweet, Visual Perception, Academic Press, New York, 1970.

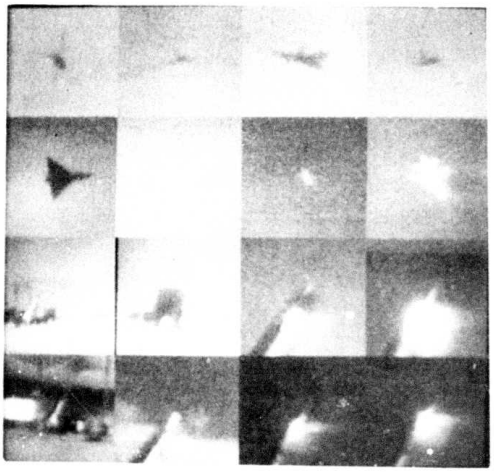


Fig. 1 Sample targets from TV video each block is 64x128, expanded to 128x128 (1 field only). The first five targets are airplanes, the next three are cruise missiles, the last eight are missiles.

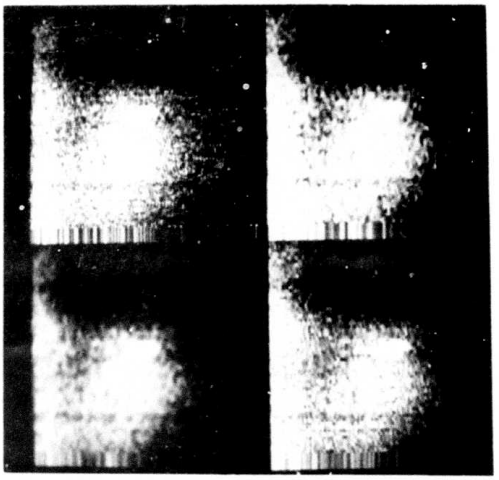


Fig. 2 Cruise missile preprocessing. Original (upper left), median filtered (upper right), averaged, median filtered (lower left), averaged only (lower right)

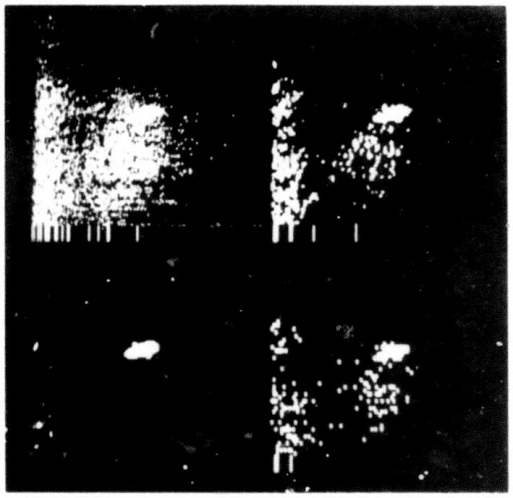


Fig. 3 Threshold versions of images in Fig. 2.

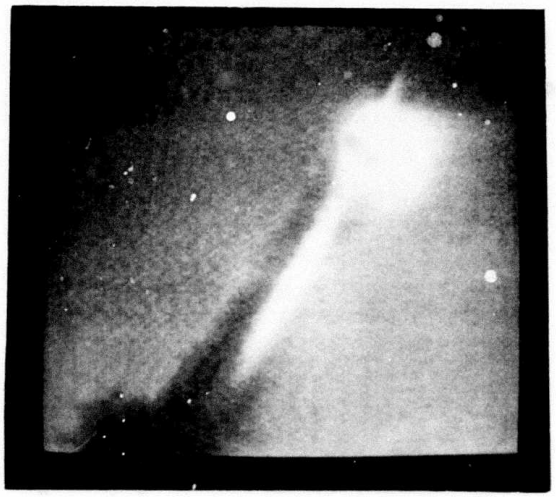


Fig. 4 Missile original, 64x128

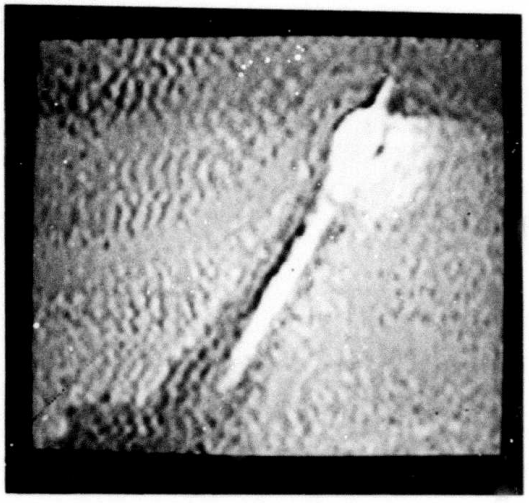


Fig. 5 Spatial bandpass filter
on Fig. 4

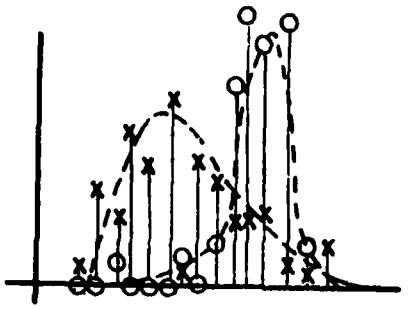


Fig. 6. Two histograms from two windows.

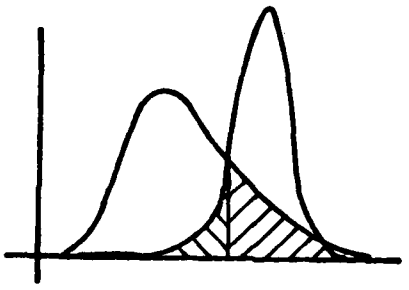


Fig. 7. Smoothed histograms and resulting overlap error.

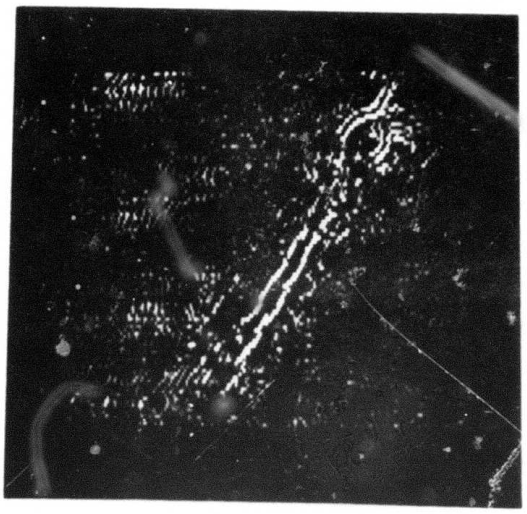


Fig. 8 Local Extrema in Fig. 5

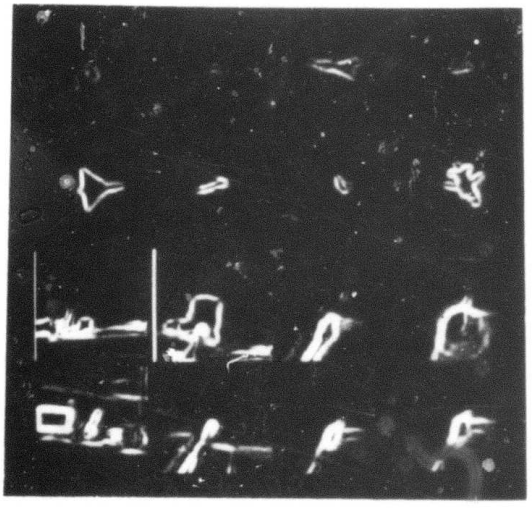


Fig. 9 Extracted edges from Fig. 1

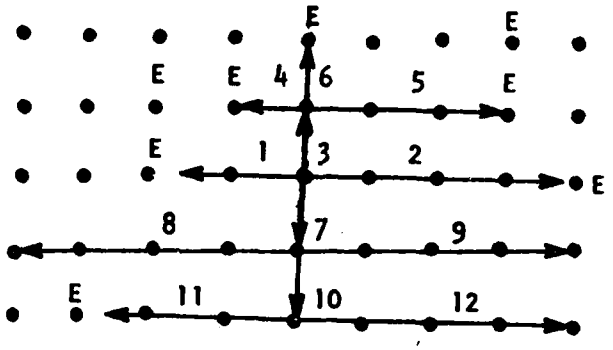


Fig. 10. Example of edge points controlling window shape.

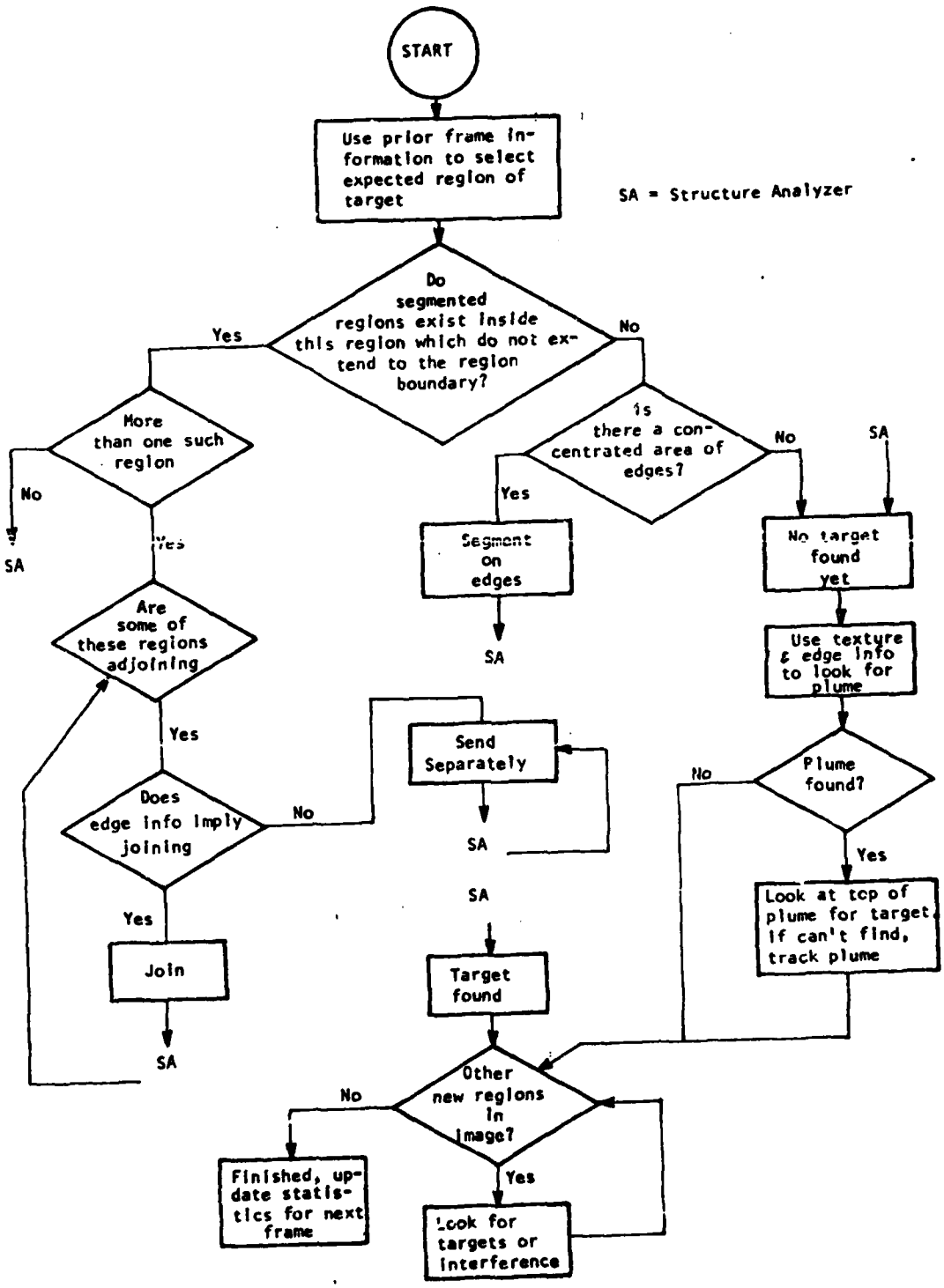


Fig. 11. Target Extraction Protocol

A SPECIAL COMPUTER ARCHITECTURE FOR IMAGE PROCESSING

K. S. Fu and Janmin Keng

Image processing by computer encompasses a wide variety of techniques and mathematical tools. In most image processing, large computers have been employed. Unfortunately, the cost is high. Image processing tasks usually involve an extremely large volume of data, much of which can be operated on in parallel. Therefore, it is important to study special computer architecture for image processing. During the last two decades the field of image processing has grown up rapidly. New techniques, algorithms, and applications have been developed, but there is still a need for improved hardware. A special computer architecture is presented here as a proposal for improving the state of the art in image processing and also to cut costs. Designing this computer architecture was a challenging problem, as the desire was to build a computer that would have the following features:

1) The computer was to allow efficient image processing at high speed utilizing interactive computation and making possible large data evaluation.
2) The computer was to preserve the general purpose aspects of a general purpose computer.
3) The computer was to be cost effective in order to allow industrial realization.

The framework of this proposed computer architecture consists of a task management processor, a parallel processor, and a sequential arithmetic processor. The task management processor is a set of software system programs serving as an operating system. The parallel processor is proposed because of the parallel nature of the operations involved in image processing. Parallel machines are considered to be particularly suitable for image processing by Thurber and Wald [7]. The parallel processor of the proposed computer.

architecture consists of an array of microprocessors which allow parallel processing capability. This parallel processor is a homogeneous system in which all processors are alike and are general purpose units. The homogeneous parallel processor lends itself quite readily to extendability as such systems are usually modularly constructed. With modular parallel processing, the system's memory, processor, and input/output modules may be enlarged as processing requirements increase; thereby avoiding replacement of the entire parallel processing system. The modular parallel processor also provides very high reliability since, with several identical modules of each types, the system can withstand failures in several modules and still operate. This arrangement also increases efficiency and through-put since all of the processors could be operating simultaneously. The sequential arithmetic processor is a microprogrammed controlled processor which performs the sequential arithmetics and also controls the input/output devices. The details of these processors will be set forth in section 2. It will be shown that the design goals were achieved through the proposed computer architecture.

PREVIOUS WORK AND COMMENTS

The previous work in the field of special purpose computer architecture for image processing basically falls into two categories: bit-plane processing and distributed processing. The bit-plane processing approach performs the arithmetic computation on the image points which are stored in Boolean bit-plane. For example, if the image has eight grey levels, then the image points are stored in three Boolean planes. The bit-plane processing approach tends to have a large number of processors which perform Boolean operations. The distributed computing approach utilizes processors which have powerful computation capability. These processors are designated by microprogram or hardware to execute certain specific tasks. Thus, this configuration forms

a distributed computing architecture. A detailed illustration of the previous special purpose computers for image processing is presented in our work [1]. The comparison of those computers is also discussed in [1].

Comparing the bit-plane processing approach and the distributed computing approach, the bit-plane approach mostly uses Boolean operators as processors. Boolean operators work only on binary images which are not common in the real world. One way to get around this is to use several binary picture planes to represent the grey scale values of picture points. But, the complex software and additional memory requirement cause another problem and this problem limits the processing power of the processor. One of the drawbacks of the bit-plane processing approach is that the processing power of the processor may not be adequate for some of the more sophisticated image processing techniques. For example, the parallel Picture Processing Machine PPM (now called PICAP) has been shown in Kruse [8] to be applicable only to the preprocessing part of fingerprint classification and as stated above syntactic techniques have to be performed by the conventional computer in the PICAP system. Thus, the feasibility of the bit-plane approach to perform highly sophisticated, but important, techniques is, at this time, unsure. The capability of real-time processing of the bit-plane processing computers such as CLIP 3, CLIP4, and PPM is very difficult to ascertain until more complicated techniques have been implemented by these computers using pictures with more grey levels such as 128 or 256.

After studying the feasibilities of the bit-plane processing and distributed computing approaches to the real world image processing task, we feel that the distributed computing approach is more feasible considering the present state of the art with respect to both software and hardware. In March 1977, Stone [6] indicated that the distributed computing approach is one of

the future trends for general computer architecture. His remark supports our judgement on the distributed computing approach for special computer architecture for image processing. A major drawback of previous computers designed by distributed computing approach is that the processor systems are not reconfigurable. The vast varieties of sensor types, applications, and image processing techniques, require that the image processing system (especially the parallel processor) be reconfigurable. Therefore, a generalized computer architecture which is reconfigurable under software control is proposed in the next section for the many applications of image processing. Not only is the concept of reconfigure-ability new for special purpose computer architectures for image processing, but also the methods of exploitation of parallelism is new. In the proposed computer architecture parallelism within the task is exploited by the parallel processor. In the meantime the operations of the sequential arithmetic processor are pipelined with the parallel processor under programmer control in certain tasks which can be decomposed into pipelined processing. Therefore, parallelism and pipelining are exploited at the same time in the proposed computer architecture. The parallelism is referred to the multiprocessing approach which subdivides each outcoming job among many identically constructed mechanisms. The piping, or overlap is referred to another multiprocessing approach which is to develop a collection of specialized mechanisms capable of working simultaneously to form a general purpose organization. The processing time of the image processing task by the proposed computer architecture will be sped up by a factor which is comparable to the amount of parallelism and pipelining existing in the image processing task of interest.

PHYSICAL ORGANIZATION AND CONTROL FLOW OF THE PROPOSED COMPUTER ARCHITECTURE

The proposed Computer Architecture for Image Processing is called CAIP and is designed using the most recent semiconductor technology. The physical organization and control flow are as follows:

Physical Organization of the Proposed Computer Architecture

The physical organization of special computer architecture for image processing (CAIP) comprises the task management processor, the control units, the parallel processor, the sequential arithmetic processor, and the memory organization.

1) Task Management Processor (TMP) is a set of software programs which allocate the jobs to the parallel processor (PP) or Sequential Arithmetic Processor (SAP). The set of software programs include a task control program, a job control program, an input/output program, and the language translation program. The task control program provides the logical interface between the hardware and the remainder of the software system and is responsible for the allocation of jobs to the parallel processor and sequential arithmetic processor. Each task has a tag which is designated by the programmer for the identification of parallel processing or sequential processing. One part of the task control program is called the tag examination program which examines the tags on tasks and allocates the tasks to the proper processor. Following the tag examination, the initiation program, which is another part of the task control program, initiates the parallel processor or the sequential arithmetic processor. In general, the task control program performs scheduling, supervision, interruption handling, execution supervision, and clock supervision. The job control program provides a logical interface between a task and a job or between a task and the system operator. The job control program analyzes the job stream, looks at system resources, processes job execution and

termination, and communicates between the system operator and the individual job program. The I/O control program provides an interface between the processing programs and the I/O devices. The I/O control program performs I/O supervision, access routine processing, and I/O device initiation. The language translator program translates the computer language into machine codes and compiles the program to be executed.

2) The Control Unit (CU) consists of two sets of software programs. One control unit (CUPP) is for the parallel processor, and the other (CUSA) is for the sequential arithmetic processor. The CUPP and CUSA are different from conventional processors, in that they are software programs which control the operation of the parallel processor and the sequential arithmetic processor respectively. The CUPP is a control program which initiates two different sets of software operating systems, one for SIMD mode and the other for MIMD mode. The two sets of software operating systems drive the parallel processor individually upon the command of CUPP. The reconfiguration from SIMD mode to MIMD mode or MIMD mode to SIMD mode are performed by loading the operating system corresponding to the desired mode. Next, the operating system is assigned to the parallel processor (PP) by the control program of CUPP. Hence, the parallel processor operates in either SIMD or MIMD modes under the respective operating systems. Through this arrangement, the CUPP reconfigurates the computer architecture from SIMD to MIMD or MIMD to SIMD. This reconfigurable capability enables this computer architecture to satisfy the large variety of applications of image processing. The operating system of MIMD mode includes scheduling routines, dynamic allocating routines, and dispatching routines. The scheduling routines schedule each job depending on job priority and facility requirements. The dynamic allocating routines take jobs set up by the scheduling routines and partition the set of processors according to the needs

of each job. The dispatching routine dispatches the processors when the job terminates or some higher priority task requires processors. The operating system for the SIMD mode is on the master control unit. All the processors of the parallel processor (PP) are controlled by this master control unit and thereby an instruction is executed simultaneously on all the processors. The control unit of the sequential arithmetic processor (CUSA) controls the sequential arithmetic processor which is a microprogram-controlled Bipolar processor. The control units are shown in Fig. 1. Note that Figs. 1, 2, 3, and Fig. 4 form a graphical illustration of this computer architecture (CAIP) by linking the corresponding symbols α, β, γ, σ in the figures.

3) The Parallel Processor (PP) is an array of microprocessors. For the parallel processor, N microprocessors are connected in an array fashion. The array organization is especially suitable for image processing [7]. The number N is determined from the tradeoff considerations between performance and cost. The optimal number, N, varies with the task. Hence, N can only be determined at the time of implementation. In the framework shown in Fig. 2, a set of 64 microprocessors is used to give an idea of the dimension of the problem. The control unit (CUPP) controls this set of microprocessors in SIMD or MIMD modes. This control unit enables the parallel processor (PP) to have a higher degree of flexibility and processing power. The SIMD mode utilizes a single master control unit which drives the multiple processing units (microprocessors), all of which either execute or ignore the current instruction. This SIMD mode is especially useful for the cases in which there exists (1) a large amount of independent data, (2) no restrictions preventing them from being processed in parallel, (3) a requirement for high throughput, and (4) a possibility of exploiting the associate addressing selection technique. Thus, SIMD mode is suitable for the local task, which executes the same instruction on each

PROPOSED ARCHITECTURE OF SPECIAL PURPOSE ARRAY PROCESSING,
IMAGE PROCESSING COMPUTER

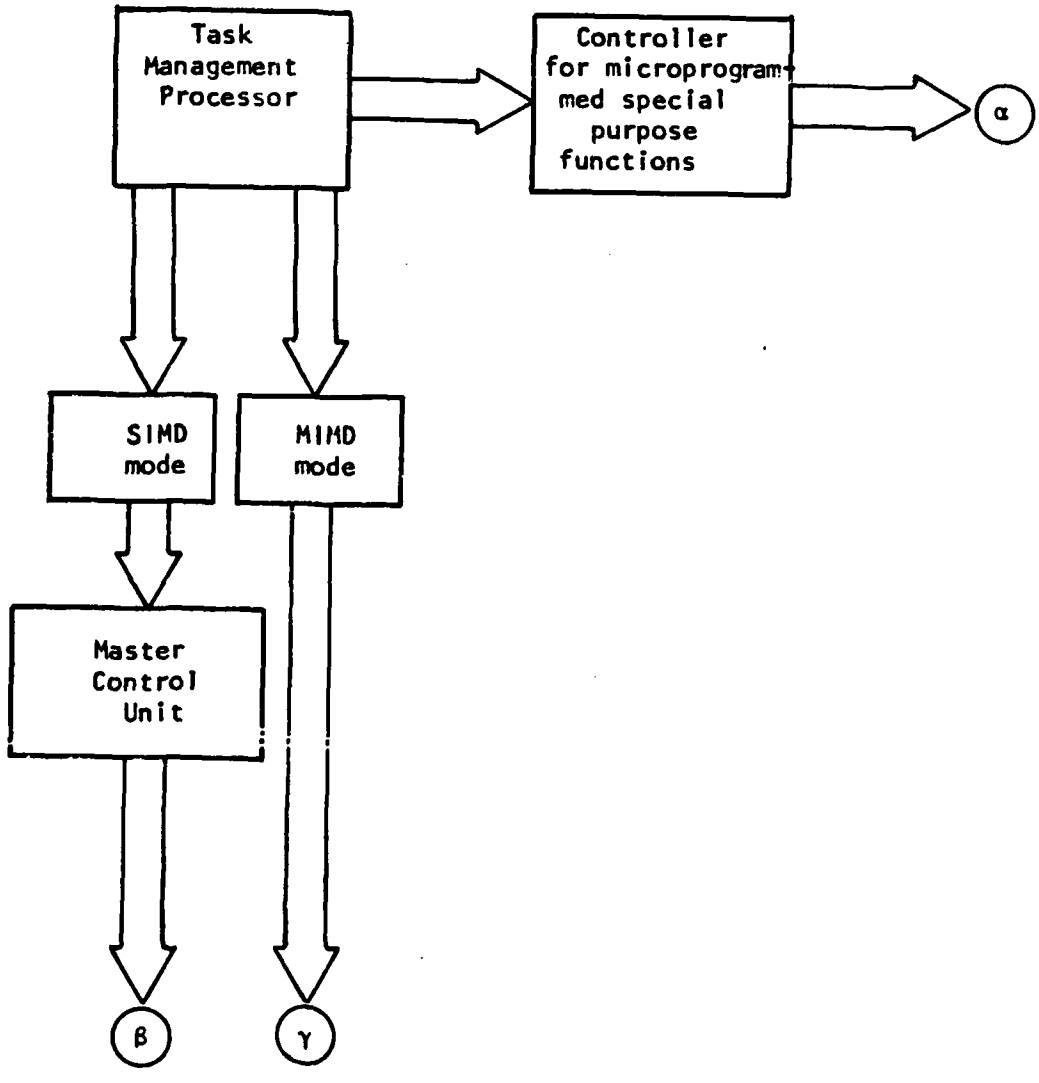


Figure 1. Control units of designed computer architecture for image processing.

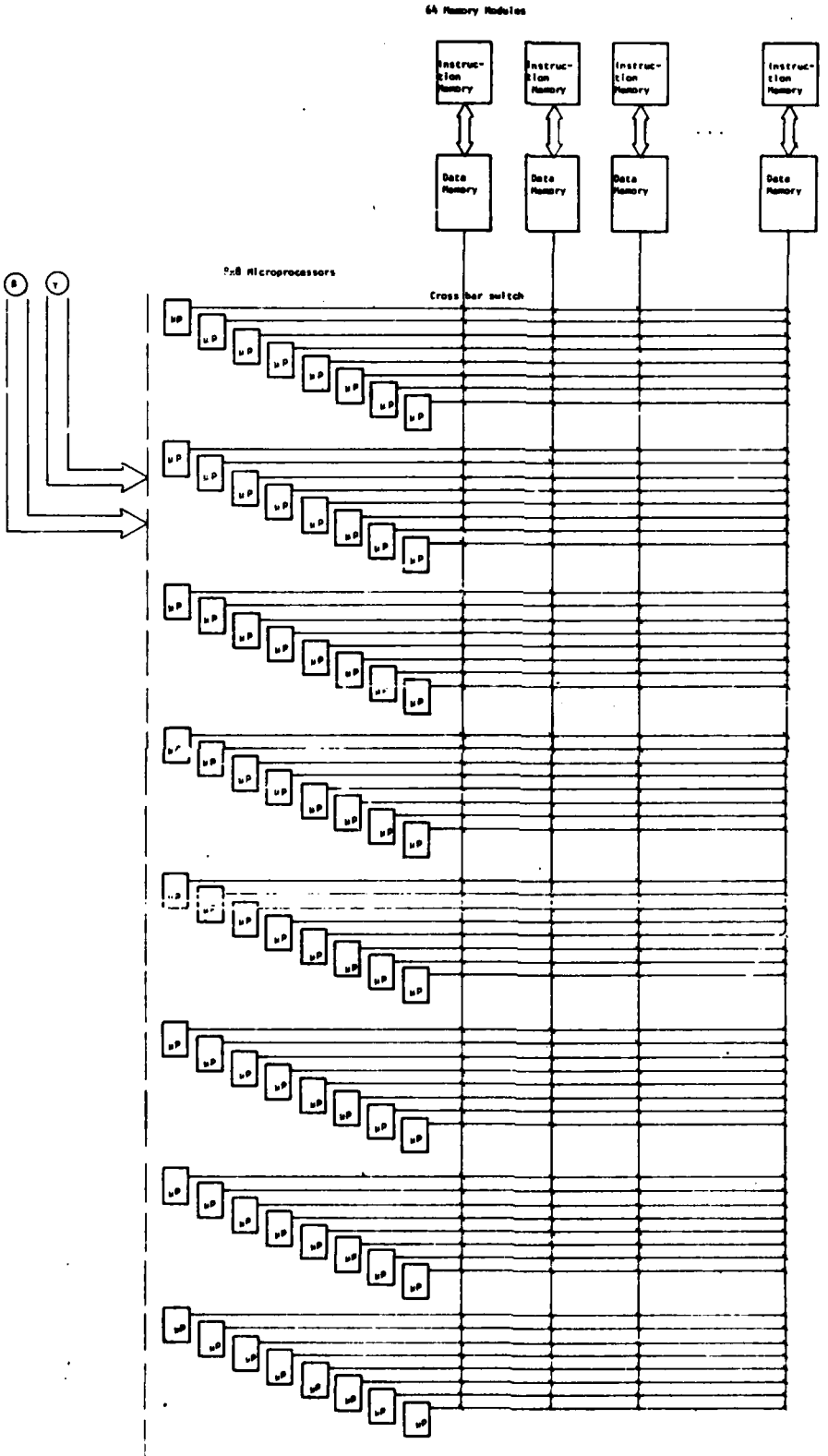


Figure 2. Parallel processor of designed computer architecture.

picture element within an image window. The MIMD mode utilizes N processors and N memories where each processor follows an independent instruction stream. The parallel processor (PP) is connected by crossbar switches to an interleaved memory system which divides the ordinary memory into modules and the consecutive data are stored in different modules. The interleaved memory system is used because the bandwidth is greater than a conventional memory system and can be multiply-accessed which is more appropriate for parallel processing than a conventional system, in which data can only be accessed one-at-a time.

4) The Sequential Arithmetic Processor (SAP) is a microprogram-controlled processor. Mini-and micro-computers are not used here because user microprogrammable capability and Bipolar processor are not furnished by the usual mini- or micro-computers. The Sequential Arithmetic Processor (SAP) is a Bipolar processor which is a processor built by Bipolar semiconductor technology and usually has the bit slicing capability. The Bipolar processor permits the designer to define his own instruction set and the associated hardware architecture to achieve special capabilities, such as, variable word length capability, or to perform an application with the highest efficiency. The Bipolar processor expands the CPU word length by cascading the needed number of bit-slice microprocessor components. This variable word length capability of SAP makes it more general and powerful than other microprocessors. Along with this processor, a microprogram memory is needed to store the microprograms of certain programs frequently in use. For the microprograms of image processing techniques, no instruction fetching is required because of the coding of the microprogram. The microprogram capability saves processing time according to the ratio of instruction fetch time to total execution time. The Sequential Arithmetic Processor is shown in Fig. 3.

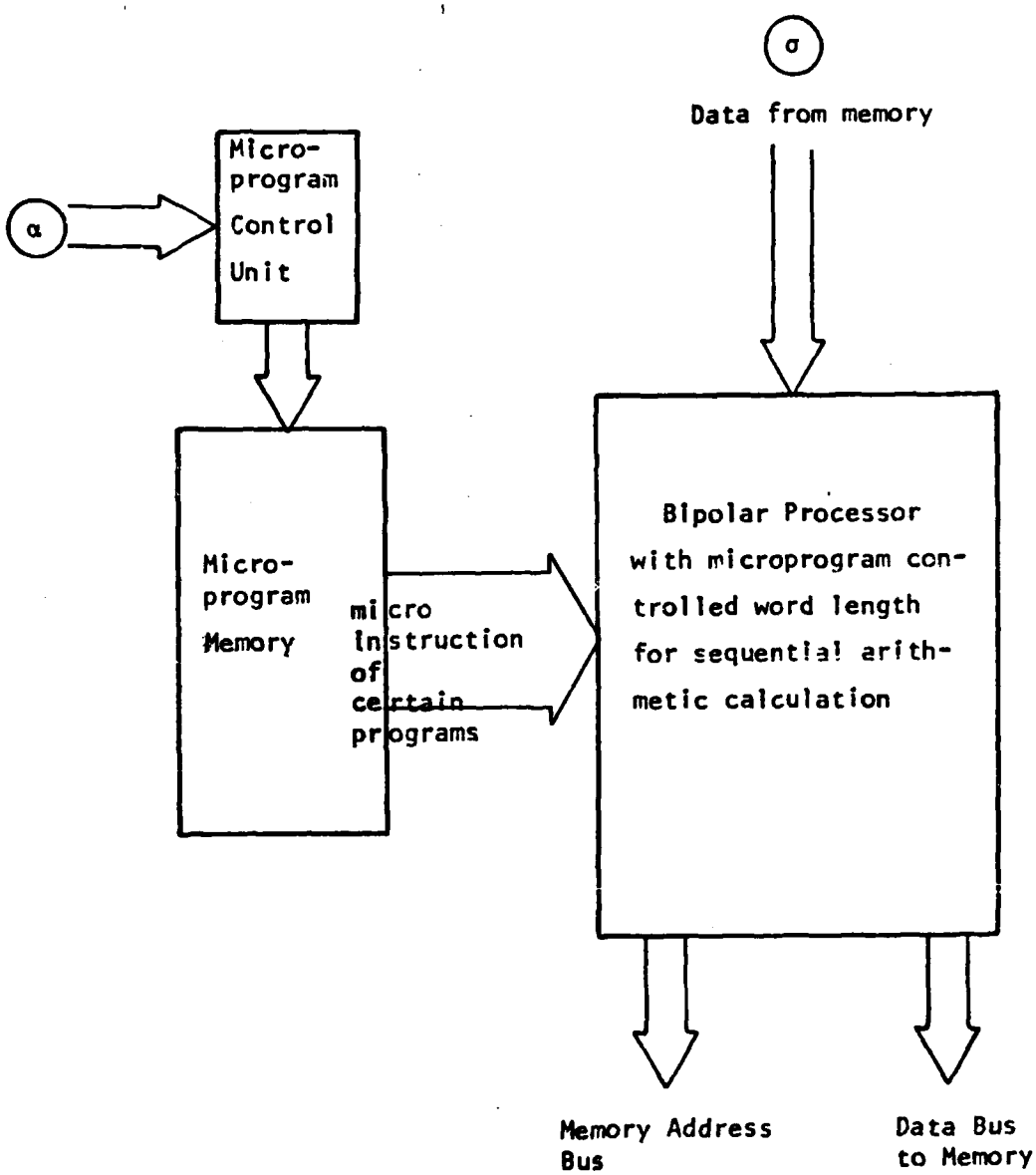


Figure 3. Sequential arithmetic processor of designed computer architecture.

5) The Memory Hierarchy Organization is the memory system for SAP. The picture is stored on a magnetic disk memory which is more economical than core memory, but memory access time is long. The memory access time of the Bipolar memory is faster by a factor of 100 than disk memory [9]. Therefore, a semiconductor Bipolar memory is connected to the disk memory as working memory space and buffer. The hierarchy organization is as follows: the picture area which is to be processed is loaded onto the Bipolar memory from the disk memory, then the processor gets the data (picture points) from the Bipolar memory thus allowing extremely fast memory access time. In order to avoid being delayed by the loading time from disk to Bipolar memory, a Bipolar Memory Buffer (BMB) is used. While the processor is reading the data into the Bipolar working memory space, the next picture area is loaded onto the Bipolar Memory Buffer (BMB). Thus, this memory preloading makes the data always ready in the fast-access Bipolar memory. This memory hierarchy organization is illustrated in Fig. 4. The proposed memory hierarchy loads a large block of data onto the Bipolar Buffer Memory as image processing necessitates operations on large blocks of data, this ability of the CAIP through its BMB provides a marked advantage for image processing. Once the large block of data has been loaded onto the Bipolar memory, any individual data point can be randomly accessed and individual loading from primary to secondary storage is not needed. Thus, the user's software becomes simpler by virtue of this proposed memory hierarchy organization. The Bipolar memory is used here because the Bipolar memory is the memory device with the fast fetch-time in present technology [9]. There is only one disadvantage to the use of Bipolar memory: It is more expensive than core memory. Instead of using Bipolar memory, core memory can be used in the proposed memory hierarchy if the cost is a great constraint to the user.

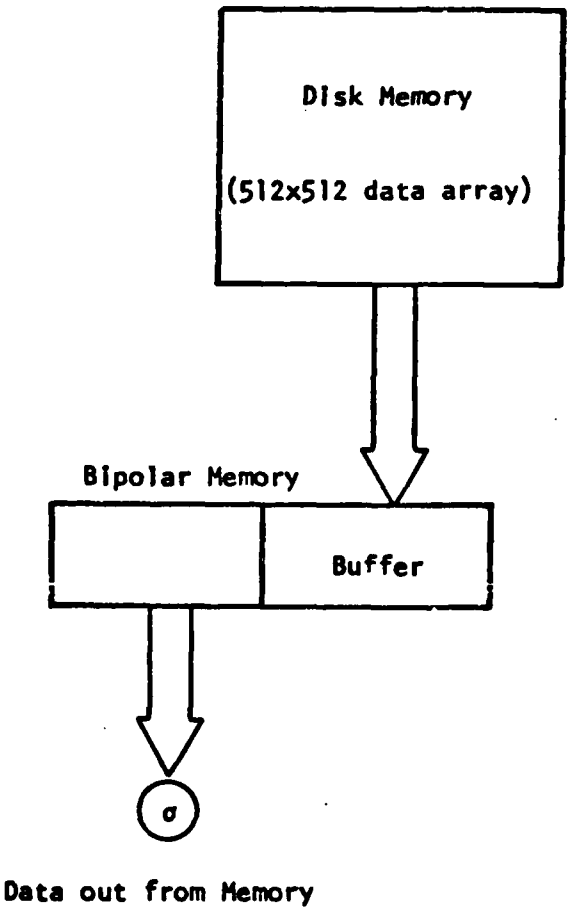


Figure 4. Memory hierarchy of designed computer architecture.

Control Flow of the Proposed Computer Architecture

The control flow for the proposed computer architecture, CAIP, is shown in Fig. 5. The Task Management Processor (TMP) allocates jobs to the parallel processor (PP) or the sequential arithmetic processor (SAP) by means of the tag examination routine. The user's program provides a flag to the parallel processor which indicates whether the task is local or global. The locality tester examines the flag and initiates the SIMD or MIMD mode. The SIMD mode is appropriate for local tasks. In this mode a single instruction is executed simultaneously on the image points. The local task means that the instruction is executed on individual data within an image window, such as calculating the histogram of an image window. For global task, the MIMD mode is employed. The global task means that part of the task is performing one kind of operation and other part of the task is performing another kind of operation. For example, in the task of evaluating textured and nontextured areas, some processors perform second order statistical texture analysis on certain window and some processors perform first order mean vector analysis on the corresponding windows. This global task comprises of two different natures of subtasks. The outputs from the Sequential Arithmetic Processor (SAP) communicates with the parallel processor. Therefore, the SAP may support the PP and vice versa. Parallelism of task is exploited by the parallel processor (PP) to obtain high speed performance. In the meantime the operations of the sequential arithmetic processor (SAP) are pipelined to the parallel processor under program control in certain tasks which can be decomposed into pipelined processing. Therefore, the control flow of this architecture exhibits both parallelism and pipelining . This arrangement has not been incorporated into any of the previously proposed systems. Since two types of multiprocessing parallel processing and pipeline processing, are exploited

CONTROL FLOW OF DESIGNED COMPUTER ARCHITECTURE

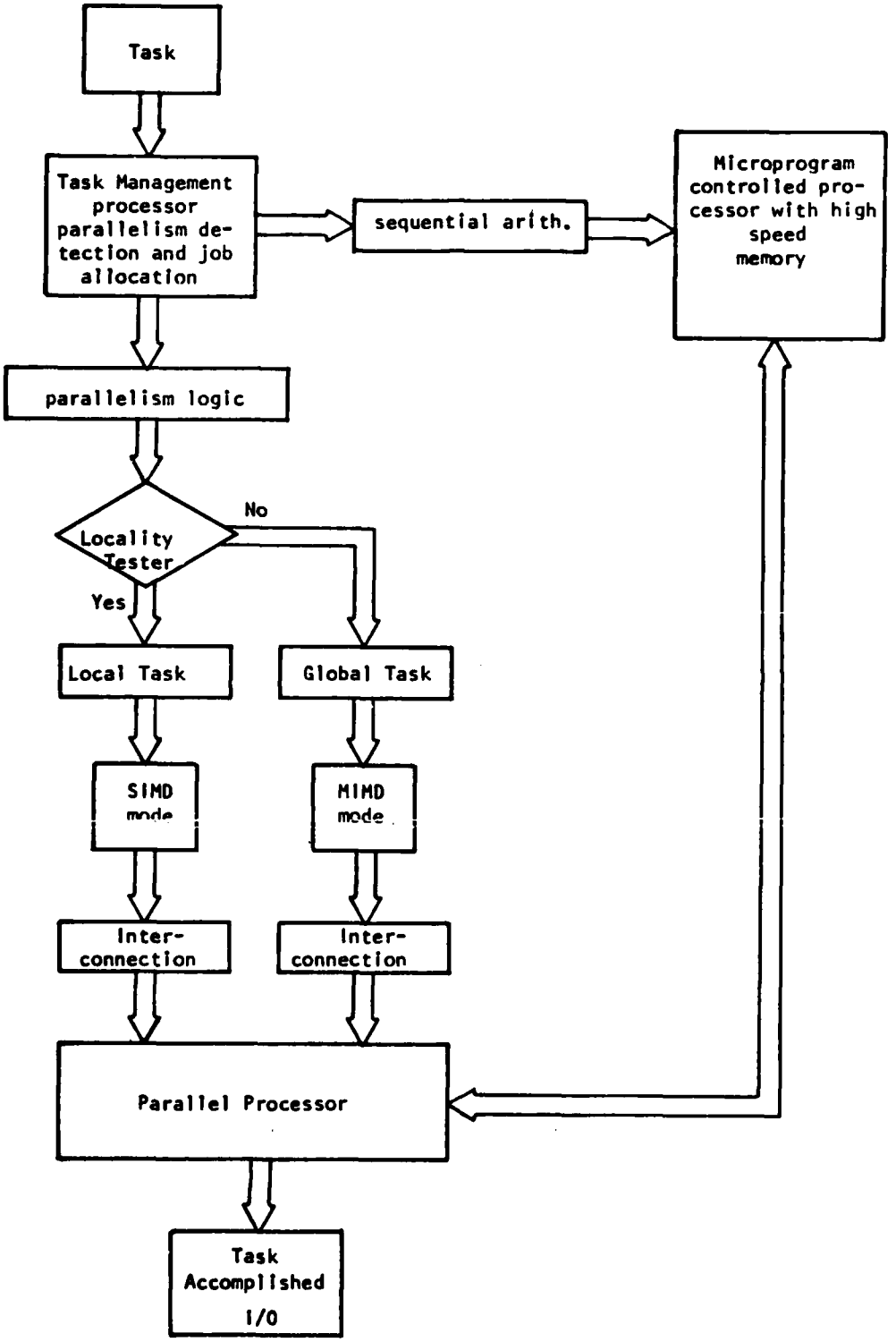


Figure 5. Control flow of designed computer architecture for image processing, array processing.

simultaneously, this contributes to high speed performance in the proposed computer architecture.

ANALYSIS

Performance and Cost-Effectiveness Tradeoffs

The parallel processor (PP) of the proposed computer architecture consists of an array of microprocessors which are the processing elements. In determining the optimal number, N, of microprocessors for the parallel processor (PP), a systematic procedure is needed. Such a procedure follows: In designing the parallel processor, as the number of processing elements (microprocessors) increases, the number of data points processed by each processor decreases and processing speed increases, but the scheduling overhead also increases. Therefore, the processing speed improvement reaches a saturation point at a certain number of processing elements. So it seems that the number of processing elements corresponding to the saturation point would be a good choice. However, the answer is not that simple, as cost-effectiveness is an important factor in the feasibility of a computer architecture. Thus, the costs, such as hardware and software costs of the parallel processor (PP), need to be considered. Hardware cost usually involves the hardware purchased. Software cost refers to development of the operating systems and software supports. The hardware cost increases with the increase in the number of processing elements (microprocessors). The software cost increases more rapidly than the hardware cost as the number of processing elements increases. The best choice of optimal number of processing elements is obtained by evaluating the performance improvement and cost increment on different image processing tasks. The optimal number directly depends on the specifications which are given by the user. For example, if the performance curve becomes saturated at 80 processors for task A, and 70 for task B shown in Fig. 6, N should be in

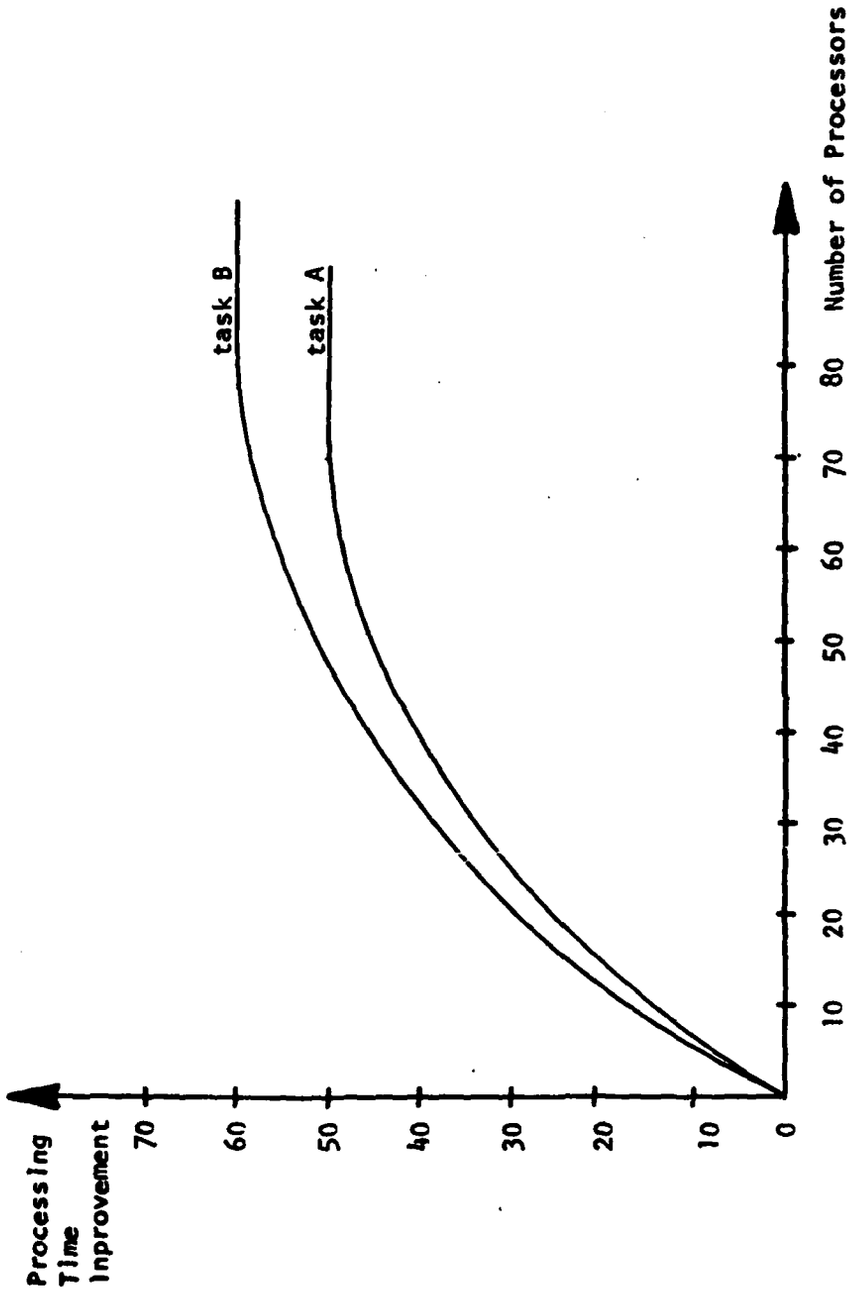


Figure 6. Performance Improvement versus Increments of number of processor.

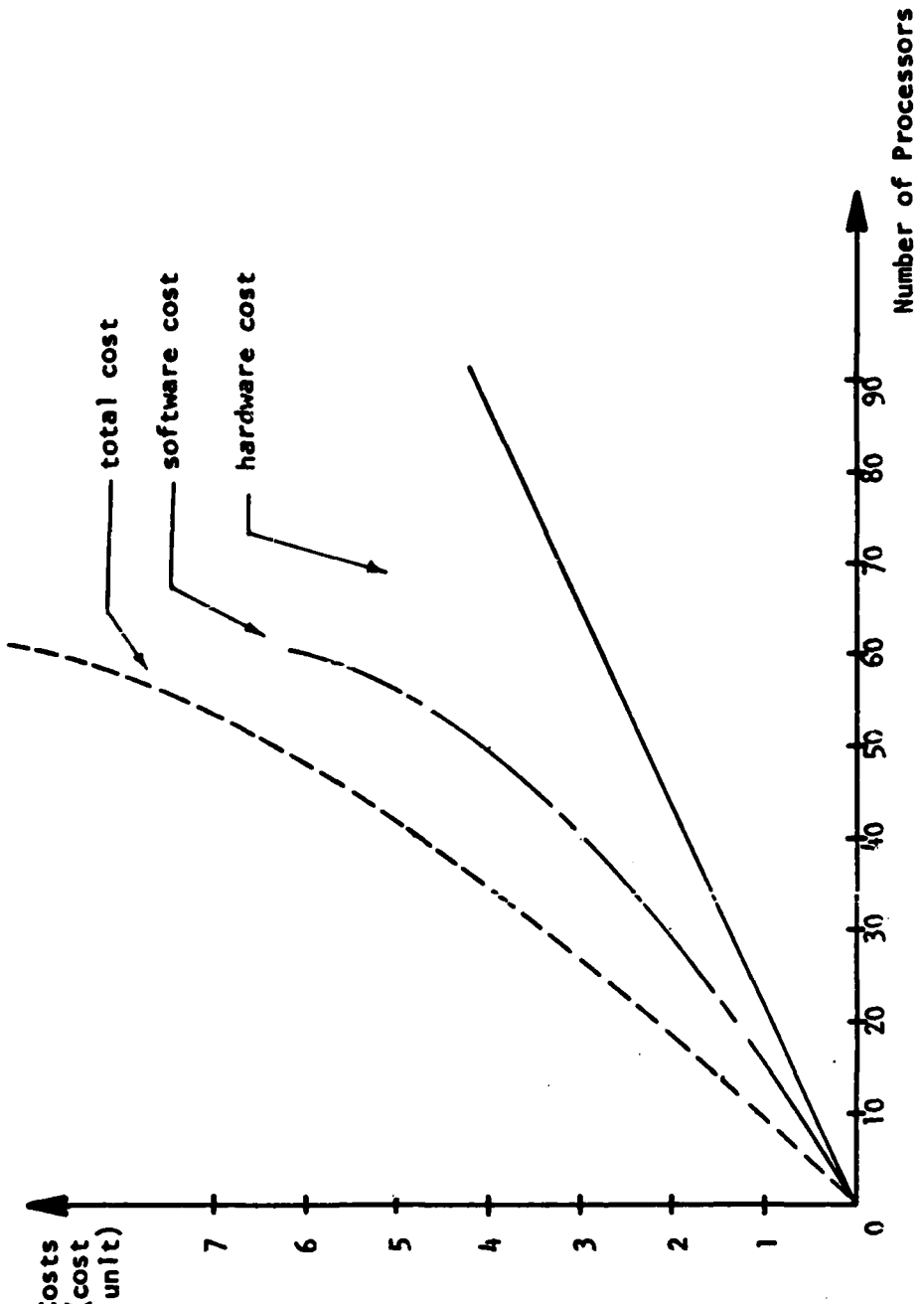


Figure 7. Cost estimation versus number of processors.

between 70 and 80. The hardware cost increases more or less linearly with the number of processors (if fewer than 100 processors are bought). But, as stated above, the software cost increases more rapidly than the hardware cost. If the software cost increases sharply at 60 processors, as in Fig. 7, the optimal number of processors, considering cost and performance trade-off, is between 60 and 80. Depending on the specifications of the user, if the concern is more for performance than cost, then a number near 80 is chosen. If the user is more concerned about cost, then a number near 60 should be chosen.

Implementation of Statistical Methods

During recent years, a number of image processing algorithms have been developed [2,3]. In this section, some image processing techniques are discussed in terms of the proposed computer architecture, CAIP, to exemplify the operations of the special computer for image processing.

Statistical texture analysis has been an important topic in the field of image processing. The texture analysis technique in [4] consists of histogram equalization, and texture feature measurement. In applying the proposed computer architecture, CAIP, to such a texture analysis technique, the task of texture analysis is assigned a tag P (which stands for parallel processor task) by the user's program. (Tag S stands for sequential arithmetic processor task). The subtasks of texture analysis, such as histogram equalization and texture feature measurement, are designated by the flags SI (which denotes the SIMD mode for the task), and MI (which denotes the MIMD mode for the task). Assuming the size of the image is MxM. The texture analysis task is allocated to the parallel processor (PP) of the CAIP by the tag examination routine of the Task Management Processor (TMP). The Control Unit of the Parallel Processor (CUPP) finds the flag, SI, for histogram equalization and loads the operating system of the SIMD mode. Each processor of the N microprocessors then

calculates the histogram of an $\frac{M}{\sqrt{N}} \times \frac{M}{\sqrt{N}}$ picture window. The outputs of the N processors are then put together to obtain the equalization result of the final histogram. The CUPP keeps the parallel processor (PP) in the SIMD mode after examining the falg SI of the texture feature measu[illegible]ent task. For example, if the 88x88 picture discussed in [4] is to be processed, each processor will process the co-occurrence matrix of the window of 11x11 pixels. The variability texture feature measurement is calculated by each processor as the texture value for the center cell (4x4) of that window. The mapping of the array of processors to the image points is shifted four pixels and repeats the texture feature measurement task. When this shift reaches the right edge of image, the mapping is shifted four pixels downward and the process is repeated from the left most column of the image. This process continues until the texture values of all the picture points are obtained.

Syntactic Methods and Parallel Processing

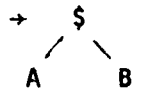
As has been pointed out previously, syntactic methods for image processing have increased in importance for certain applications. Previous special computers such as the PPM and the PICAP are unable to process syntactic methods by parallel processing [8]. However, the proposed CAIP furnishes the capability of parallel processing of syntactic algorithm. In using the parallel processor for syntactic methods, parallel parsing schemes are most desirable as they can utilize the capability of the parallel processor. Unfortunately, the research in parallel parsing schemes is very limited. In this section, we introduce the parallel parsing of tree languages and explore the parallel parsing of the parallel context-free languages [5].

1) Tree Languages and Parallel Processing. The parallel processor (PP) of the proposed computer architecture for image processing can be applied to tree grammar parsing. The parallel parsing procedure of a tree grammar is described below. The task of tree grammar parsing is designated by a flag P and tag SI

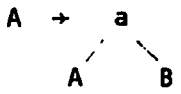
by the user's program for the effective utilization of the facilities. Through the control unit of the computer architecture, the parallel processor is put into the SIMD made for the task of tree grammar parsing. If a production rule of the tree grammar which is applied to parse the language has k branches, then each of these k branches has a nonterminal. Each processor of the parallel processor (PP) is assigned by the user to parse one nonterminal. This procedure is applied to consecutive parsing of the language until a final parsing result is achieved. If the parsing is successful, then the language (pattern) is accepted. If the parse fails then the language (pattern) is rejected.

For example, the tree grammar is $G_t = (V, r, P, S)$, where $V = \{S, a, b, \$, A, B\}$, $V_T = \{\rightarrow a, \rightarrow b, \$\}$, $r(a) = \{2, 1, 0\}$ $r(b) = \{2, 1, 0\}$, $r(\$) = \{2\}$ and P: [2]

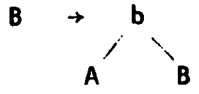
rule 1:



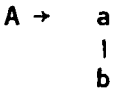
rule 2:



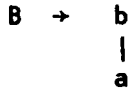
rule 3:



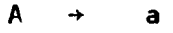
rule 4:



rule 5:

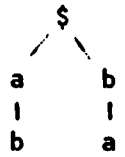


rule 6:

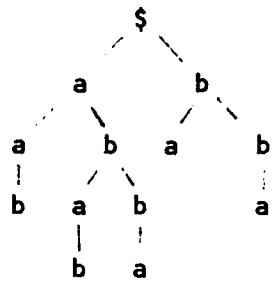


This grammar generates such patterns as

(1)



(2)

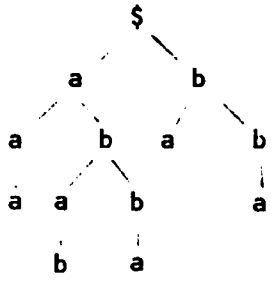


In order to perform parallel parsing of the tree language, processors need to be assigned. First, the depth "d" of the tree is defined as the number of the levels of the tree. The depth of the tree in (1) is 2 (d=2) and the depth of the tree in (2) is 4 (d=4). The maximum number of branches for all the tree grammar rules is easily obtained by checking the values of r in the grammar $G_t = (V, r, P, S)$ and this number is called m. The relationship between m and r is that m is the maximum of the values of the r's. The number of needed processors in the parallel processor (PP) is $(d)^m$. This procedure is performed for the worst case protection concept which allows the maximum number of branching in each level of the tree parsing. If the number $(d)^m$ is greater than the number of processors of the parallel processor (PP), there are two solutions: One is the static priority procedure, and the other is the dynamic priority procedure. The static priority procedure has a fixed priority rule for assigning the available processors to the proper subtasks. The fixed priority rule for the parallel tree parsing scheme is to assign the k left most branches the equal priority in each parsing stage. The number, k, is the number of available processors of the parallel processor (PP) for each parsing stage. At parsing stage one, the k left most branches are parsed first, based on the highest priority rule. At parsing stage two, the k left most branches (nonterminals) of stage two then have the highest priority to be parsed. Thus, the k available processors are assigned to parse these nonterminals.

In the dynamic priority procedure, a dynamic priority rule has to be established at each stage of parsing in order to determine which subtasks have the highest priority. For example, the k left most priority could be assigned first, then the k rightmost priority assigned next at the request of the user. However, in parsing the tree languages, all the individual branches (non-terminals) have to be parsed to get the nodes (terminals), therefore, the static priority procedure is better. The dynamic priority procedure would only be used in special cases, such as the case in which only a partial parsing result is of interest.

Using the example given above to illustrate the proposed parallel parsing scheme for tree languages and to compare the parsing result with conventional parsing scheme for tree languages, if the depth of the tree to be parsed is, at most, four - the maximum number of processors needed is easily calculated to be $(4)^2=16$. In the parallel parsing of the input tree α.

$\alpha =$



The task is allocated to the parallel processor (PP) by the P flag found by the Task Management Processor (TMP) and the parallel processor (PP) is placed in the SIMD made by CUPP. The procedure is graphically illustrated in Fig. 8. At the parsing of depth one, one processor parses the language by the rule 1. At the parsing of depth two, two processors are assigned. The number of processors needed for the parsing of depth k is automatically determined by the number of branches obtained from the parsing of depth k-1. The number of branches obtained from depth one in our example is two. Thus, two processors

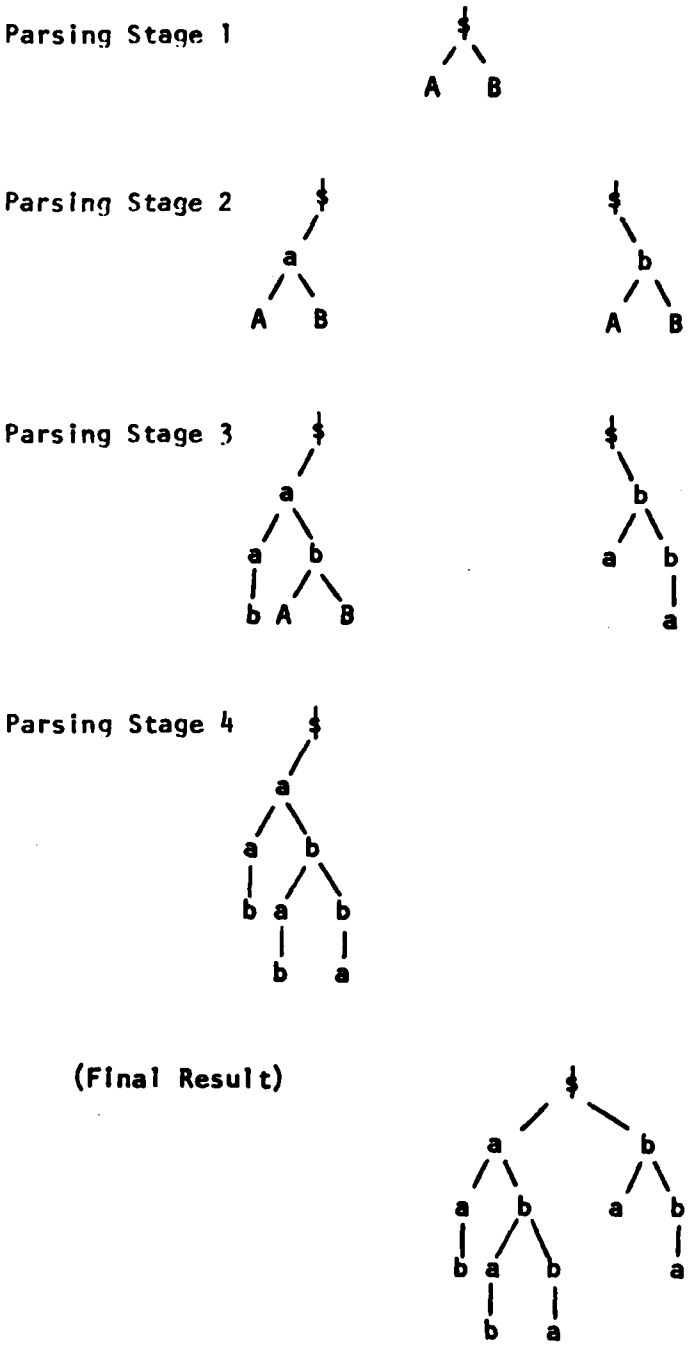


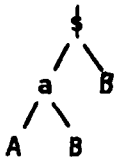
Figure 8. Parallel tree parsing procedure.

are needed, one for the parsing of the branch starting from nonterminal A and the other for the parsing of the branch starting from nonterminal B. Grammar rules 2 and 3 are simultaneously applied by the two processors to parse the two branches of the tree α starting from A and B respectively. At the parsing of depth three, four processors are needed. Two processors simultaneously parse the branches of A and B which are the result of parsing rule 2 in the depth two. The parsing rules for these two processors are rules 4 and 3, respectively, to nonterminal A and B. The other two processors simultaneously apply rules 6 and 5 to parse the branches of A and B, which are the result of parsing by rule 3 in the depth two. At the parsing of depth four, since there are only two nonterminals A and B left from the parsing of depth three, two processors are assigned to simultaneously parse the branches A and B by rules 4 and 5 respectively. Thus, the parallel parsing is completed and the tree is accepted. The parsing of the same tree language by the conventional sequential parsing scheme is shown in Fig. 9. At each parsing stage, only one nonterminal can be parsed by the parser. At parsing stage one, grammar rule 1 is applied. Rule 2 is applied at parsing stage two. Then the rules 4,3,4,5,3, 6, and 5 are applied to parsing stages 3,4,5,6,7,8, and 9 respectively. Nine parsing stages are needed for the parsing of the same tree as shown in Fig. 9. It can be seen in Fig. 8 that only four parsing stages were needed for parallel tree parsing scheme. Thus, in this example there is a saving of over 50% in parsing stages, and, therefore, a corresponding saving in time by utilizing this parallel parsing scheme on the proposed computer.

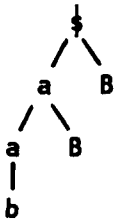
2) Parallel Context-Free Language and Parallel Processing. The parallel context-free language was defined by Siromoney and Krithivason in [5]. The definition of a parallel context-free language is a language generated by a context-free grammar in whcih the manner of applying the grammar rules is restricted as

Parsing Stage 1

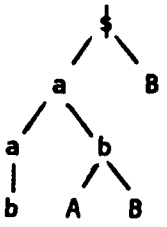
Parsing Stage 2



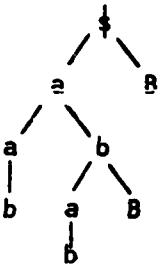
Parsing Stage 3



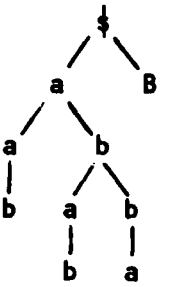
Parsing Stage 4



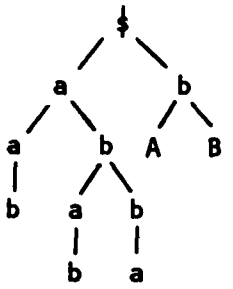
Parsing Stage 5



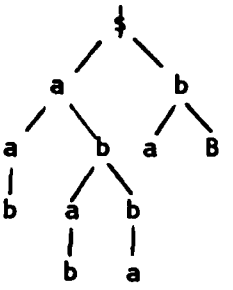
Parsing Stage 6



Parsing Stage 7



Parsing Stage 8



Parsing Stage 9
(Final Result)

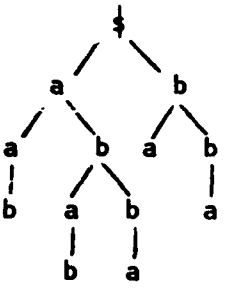


Figure 9. Conventional tree parsing procedure.

follows: if a nonterminal occurs more than once in a sentential form, then every occurrence of the nonterminal is replaced at the same time by the same rule.

The parallel processor (PP) of the proposed computer architecture for image processing (CAIP) performs the parsing of a parallel context-free language in the SIMD mode. From the definition of a parallel context-free language, the number of needed processors of the parallel processor is equal to the maximum number of occurrences of a nonterminal in all sentential forms. A processor is assigned under programmer control to each nonterminal when it occurs simultaneously with the same nonterminal in the derivation. These processors perform the parsing of same grammar rules on these nonterminals.

For example, the task of parsing a parallel context-free language is assigned the flag P and tag SI which initiate the SIMD mode for the task. The parallel context-free language is $L(G) = \{ a^{2n} \mid n \geq 0\}$ [5]. The parallel context-free grammar is $G = (V, I, P, S)$ where $V = \{ S, a\}$, $I = \{a\}$, and $P\{ S \rightarrow SS, S \rightarrow a\}$. If the languages to be parsed are aa and aaaa. The maximum number of occurrences of nonterminals in all sentential forms is four which comes from aaaa (nonterminal SSSS). Thus, the number of needed processors is four. At the first stage of parsing of the language aaaa, one processor is assigned to parse the language and the grammatical rule is $S \rightarrow SS$. At the second stage of parsing, two processors apply the same rule $S \rightarrow SS$ on the two nonterminals "S" and the parsing result is SSSS. At the third stage of parsing, four processors apply the same rule $S \rightarrow a$ on all the four nonterminals "S". Hence, the parsing result is aaaa. This language cannot be parsed sequentially as the language is defined to be parsed only parallelly. The sentence aaaa of this example is parsed by the parallel context-free grammar G. Thus, the sentence is accepted as a member of L(G).

SUMMARY AND REMARKS

A computer architecture for image processing (CAIP) has been proposed. This computer architecture is designed by the distributed computing approach. This computer comprised of a parallel processor (PP) and a sequential arithmetic processor (SAP). The reconfiguration capability of the CAIP provides the flexibility of this computer architecture and the method of exploitation of task parallelism gives the high performance of this computer architecture.

This computer architecture for image processing is proposed to use microprocessors as the processing elements. The advantage of using a microprocessor array is that the cost of microprocessors is much lower than that of conventional processors. The disadvantage is that the processing power of microprocessors is less than that of conventional processors, especially in addressing capability. For example, the most popular microprocessors INTEL 8080 and MOTOROLA 6800 do not have associate addressing or microprogramming abilities. For special image processing computers, some processing powers of conventional processors, such as associate addressing, are not essential [7]. The approach of designing a general purpose computer by microprocessors has been controversial. But, in designing a special purpose computer for image processing, microprocessors have their advantages. Furthermore, the advance in semiconductor technology is toward the development of microprocessors with higher processing power. The recent developments in the microprocessors series INTEL 3000, MOTOROLA M2900, and Texas Instruments 745481 have provided microprogramming capability to microprocessors.

With the fast growth of image processing and its applications, the need for a special image processing machine such as the proposed computer, CAIP, should certainly be appreciated.

REFERENCES

[1] Janmin Keng and K. M. Fu,"Syntactic Algorithms for Image Segmentation and Special Computer Architecture for Image Processing," Technical Report of School of Electrical Engineering, Purdue University, West Lafayette, Indiana, TR-EE 77-39, 1977.

[2] K. S. Fu, Syntactic Methods in Pattern Recognition, Academic Press, New York, 1974.

[3] A. Rosenfeld and A. C. Kak, Digital Picture Processing, Academic Press, New York, 1976.

[4] Janmin Keng, "A syntactic method for image segmentation," Proceedings of Seventh Annual Symposium of Automatic Imagery Pattern Recognition, Electronic Industrial Association, College Park, Maryland, pp. 62-87, May 23-24, 1977.

[5] R. Siromoney and K. Krithivasan, "Parallel context free languages," Information and Control, vol. 24, pp. 215-255, December 1975.

[6] H. S. Stone, "Program chairman's remark," Proceedings of Fourth Annual Symposium on Computer Architecture, March 23-25, 1977.

[7] K. J. Turber and L. D. Wald, "Associate and parallel processor," Computer Surveys, vol. 7, No. 4, pp. 215-255, Dec. 1975.

[8] B. Kruse, "The PICAP picture processing laboratory," Third Int. Joint Conference on Pattern Recognition, Coronado, California, Nov. 1976.

[9] H. S. Stone, ed., Introduction to Computer Architecture, Science Research Associates, Inc. 1975.

FACILITIES

QTY	Manufacturer	Description
3	Beehive Elect.	"Super-Bee" Terminals
2	Tex. Inst.	"Silent 700" Terminals
1	Digi-Data	Industry standard magnetic tape system; 2, 9-track and 1, 7-track drives; one each NRZI and phase-encoded formatters/controllers
1	DEC	Dual-drive DEC tape unit
1	DEC	RP03 disk drive (40 million characters)
1	Fabritek	96K-word auxiliary memory system (64K bought by ARPA, 32K by NASA)
1	Versatek	Electrostatic matrix printer
1	Comtal	Color picture display
1	Data Printer	132 column, 600 L.P.M. line printer
1	True-Data	Punched card reader
1	Tektronix	Model 4010, graphics display
1	DEC	PDP 11/45 computer system, system includes: 32K memory FPP-11 floating point processor (NSF money) H960 extension mounting cabinet 3 - small peripheral mountings blocks (DD-11) 1 UNIBUS repeater/expander DH11, 16-line terminal multiplexor KW11-p programmed clock

PUBLICATIONS

Book or Book Chapters

FU, K.S., "Some Multidimensional Grammar Inference Methods," in Pattern Recognition and Artificial Intelligence, ed. by C. H. Chen, Academic Press, 1977.

FU, K.S., "Tree Languages and Syntactic Pattern Recognition," in Pattern Recognition and Artificial Intelligence, ed. by C. H. Chan, Academic Press, 1977.

GONZALEZ, R. and WINTZ, P. A., Digital Image Processing, Addison-Wesley Publishing Co., Inc. 1977.

WINTZ, P.A. and HABIBI, A., "Image Coding by Linear Transformation," DATA COMPRESSION, ed. by L.D. Davisson and R.M. Gray, Halsted Press.

JOURNAL PUBLICATIONS

FU, K.S., "A Note on the K-tail Method of Tree Grammar Inference," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-7, April 1977. (with J.M. Brayer)

FU, K.S., "Nonparametric Bayes Risk Estimation for Pattern Classification," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-7, September 1977. (with Z. Chen)

FUKUNAGA, K. and S. Ando, "The Optimum Nonlinear Features for a Scatter Criterion in Discriminant Analysis," IEEE Trans. on Information Theory," Vol. IT-23, pp. 453-459, July 1977.

FUKUNAGA, K., "A Branch and Bound Algorithm for Feature Subset Selection," IEEE Trans. on Computers, Vol. C-26, pp. 917-922, September 1977. (with P. M. Narendra)

HUANG, T.S., "Transferring of Information from Oblique Aerial Photos to Maps," (with I. Scollar, B. Weidner, and G.Y. Tang), in Digital Image Processing, ed. by H. Nagel, Springer-Verlag, 1977.

HUANG, T.S., "Subjective Effects of Pictorial Noise," (with O. R. Mitchell), Photographic Science and Engineering, May/June 1977.

HAUNG, T.S., "Computer Image Processing in Archaeology," Umschau in Wissenschaft und Technik, Fall 1977 (in German).

HUANG, T.S., "Coding of Two-Tone Images," to appear in IEEE Trans. on Communications.

HUANG, T.S., "Image Mensuration by Maximum A Posteriori Probability Estimation," (with J. Burnett), to appear in J. Opt. Soc. of America.

MITCHELL, O.R., "A Max-Min Measure for Image Texture Analysis," (with C.R. Myers, and W.A. Boyne), IEEE Trans. on Computers, Vol. C-26, pp. 408-414, April 1977.

MITCHELL, O.R., "Subjective Effects of Pictorial Noise," (with T.S. Huang), Photographic Science and Engineering, Vol. 21, pp. 129-136, May 1977.

MITCHELL, O.R., "Filtering to Remove Cloud Cover in Satellite Imagery," (with E.J. Delp, and P.L. Chen), IEEE Transactions on Geoscience Electronics, Vol. GE-15, pp. 137-141, July 1977.

MITCHELL, O.R., "Hemispheric Specialization and Cognitive Development," (with G.H. Wheatley, R.L. Frankland), to appear in the Journal for Research in Mathematics Education.

MITCHELL, O.R., "Image Segmentation Using a Local Extrema Texture Measure," (with S.G. Carlton), to appear in Pattern Recognition.

SWAIN, P.H., "The Decision Tree Classifier: Design and Potential," (with H. Hauska), IEEE Trans. Geoscience Electronics, Vol. GE-15, pp. 142-147, July 1977.

SWAIN, P.H., "Advancements in Machine-Assisted Analysis of Multispectral Data for Land Use Applications," Proc. Fourth Symposium on Machine Processing of Remotely Sensed Data, pp. 336-343, Purdue University, West Lafayette, Indiana, June 1977.

SWAIN, P.H., "A Versatile Classifier Model for Multiobservational Analysis," (short paper), Fourth Symposium on Machine Processing of Remotely Sensed Data, Purdue University, West Lafayette, Indiana, June 1977.

CONFERENCES

Fu, K.S. and YU, T.S., "Contextual Pattern Classification of Remotely Sensed Multispectral Data," Proc. 8th Annual Systems Modeling and Simulation Conference, University of Pittsburgh, April 1977.

FU, K.S., "Inference of High Dimensional Grammars," Workshop on Pattern-Directed Inference Systems, Honolulu, Hawaii, May 23-27, 1977.

FU, K.S., and Lu, S.Y., "Error-Correcting Tree Automata for Syntactic Pattern Recognition," Proc. 1977 IEEE Conference on Pattern Recognition and Image Processing, Troy, N.Y., June 1977.

FU, K.S., "Syntactic Approach and Image Processing," Proc. 1977 IEEE Conference on Pattern Recognition and Image Processing, Troy, N.Y., June 1977.

FU, K.S., "Application of Pattern Recognition to Environmental Data Interpretation and Modelling," Proc. IFIP Working Conference on Modelling and Simulation of Land, Air and Water Resources Systems, Ghent, Belgium, Aug. 30 - Sept. 2, 1977.

FUKUNAGA, K. and SHORT, R.D., "Generalized Clustering for Problem Localization, IEEE Computer Society Conference on Pattern Recognition and Image Processing, Troy, New York, June 1977.

HUANG, T.S., "Computer Simulation of Buffer Behaviour in Runlength Coding," presented at the 1977 Picture Coding Symposium, Aug. 29-31, Tokyo.

HUANG, T.S., "Image Modeling with Applications to Measurement," (with J. Burnett), presented at the 1977 International Symposium on Inf. Theory, Ithaca, N.Y. Oct. 10-13, 1977.

MITCHELL, O.R., and CARLTON, S.G., "Image Segmentation Using Texture and Grey Level," Proceedings of the IEEE Conference on Image Processing and Pattern Recognition, pp. 387-391, Troy, N.Y., June 6-8, 1977.

STAFF

CO-PRINCIPAL INVESTIGATORS

T. S. Huang
K. S. Fu

PROFESSORIAL

K. Fukunaga
O. Mitchell
A. Reeves
P. Swain
P. Wintz

GRADUATE RESEARCHERS

S. Berger
S. Carlton
N. Chang
X. Dang
E. Delp
J. Keng
E. Kit
C. Miller
B. O'Connor
A. Salahi
G. Tang
W. Tsai
T. Wallace
G. Yang
M. Yoo
K. You
T. Yu

STAFF ENGINEER

J. Besemer

STUDENT ENGINEER

R. Rindfuss

ELECTRONIC TECHNICIANS

P. Crane
J. Rogers

SECRETARY

M. Barbour